THE UNIVERSITY *of* EDINBURGH

Towards a Level Set Reinitialisation Method for Unstructured Grids

William V Edwards

Doctor of Philosophy

THE UNIVERSITY OF EDINBURGH

September 2011

Abstract

Interface tracking methods for segregated flows such as breaking ocean waves are an important tool in marine engineering. With the development in marine renewable devices increasing and a multitude of other marine flow problems that benefit from the possibility of simulation on computer, the need for accurate free surface solvers capable of solving wave simulations has never been greater.

An important component of successfully simulating segregated flow of any type is accurately tracking the position of the separating interface between fluids. It is desirable to represent the interface as a sharp, smooth, continuous entity in simulations. Popular Eulerian interface tracking methods appropriate for segregated flows such as the Marker and Cell Method (MAC) and the Volume of Fluid (VOF) were considered. However these methods have drawbacks with smearing of the interface and high computational costs in 3D simulations being among the most prevalent.

This PhD project uses a level set method to implicitly represent an interface. The level set method is a signed distance function capable of both sharp and smooth representations of a free surface. It was found, over time, that the level set function ceases to represent a signed distance due to interaction of local velocity fields. This affects the accuracy to which the level set can represent a fluid interface, leading to mass loss. An advection solver, the Cubic Interpolated Polynomial (CIP) method, is presented and tested for its ability to transport a level set interface around a numerical domain in 2D. An advection problem of the level set function demonstrates the mass loss that can befall the method.

To combat this, a process known as reinitialisation can be used to re-distance the level set function between time-steps, maintaining better accuracy. The goal of this PhD project is to present a new numerical gradient approximation that allows for the extension of the reinitialisation method to unstructured numerical grids. A particular focus is the Cartesian cut cell grid method. It allows geometric boundaries of arbitrary complexity to be cut from a regular Cartesian grid, allowing for flexible high quality grid generation with low computational cost.

A reinitialisation routine using 1^{st} order gradient approximation is implemented and demonstrated with 1D and 2D test problems. An additional area-conserving constraint is introduced to improve accuracy further. From the results, 1^{st} order gradient approximation is shown to be inadequate for improving the accuracy of the level set method. To obtain higher accuracy and the potential for use on unstructured grids a novel gradient approximation based on a slope limited least squares method, suitable for level set reinitialisation, is developed.

The new gradient scheme shows a significant improvement in accuracy when compared with level set reinitialisation methods using a lower order gradient approximation on a structured grid. A short study is conducted to find the optimal parameters for running 2D level set interface tracking and the new reinitialisation method. The details of the steps required to implement the current method on a Cartesian cut cell grid are discussed. Finally, suggestions for future work using the methods demonstrated in the thesis are presented.

Declaration

I hereby declare that the research recorded in this thesis and the thesis itself was composed and originated entirely by myself in the Institute for Energy Systems at The University of Edinburgh.

William V Edwards

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The advent of digital computing in the 2nd half of the 20th Century has had a profound impact on the way science and engineering methods have developed. The most common tool that engineers across the globe currently use is the computer. Not only have computers replaced typewriters, calculators and drawing boards but modern engineering design methods regularly take advantage of computer programs that simulate the relevant real-world physics of a problem. This Thesis is based in one such area of technology, Computational Fluid Dynamics (CFD), the study of fluid flow using computers.

For millennia the development cycle of engineering projects has been one of build, test, fix. Practical experience learnt from real-world projects both, successful and unsuccessful, used to suffice as experimental data. As a result it is no surprise that a widely accepted engineering approach was to introduce new designs with extremely conservative factors of safety. Engineers did not have a detailed understanding of the limits their structures and machines had to withstand. Thus we have massively over-engineered masterpieces such as the Forth Rail Bridge and the Empire State Building. However, as humanity and science have progressed, the rate of innovation and understanding has accelerated. By the 20th Century, the scale of the engineering projects being attempted necessitated the undertaking of a more thorough design process by calculation and experiment.

Computational fluid dynamics (CFD), the field that has arisen from the use of computers for the study of the behaviour of fluids, originally received a large amount of attention in the fields of detonation shock waves and aerodynamics. A need for an alternative to experiments was created when a treaty on the ban of testing nuclear weapons was signed by countries, including the United States of America, in 1963. The need for computer modelling the effects of nuclear detonations became strong because there were no other options through which to obtain data.

In the same decade, as jet aircraft became faster, transonic flow regimes were impossible to recreate in an experiment as no wind tunnel can produce these speeds. It was therefore of critical importance that computer based study and design of problems was developed to step in and provide reliable data that engineers could use to optimise the design of their aircraft.

At the beginning of the 21st Century the marine engineering sector is faced by similar difficulties to those of the aerospace sector in the ‘60s and ‘70s. Engineers in the marine sector are seeking solutions to estimating loads on exposed structures placed in the marine environment. What approach should be taken to the validation of new designs without committing to building expensive full size prototypes and deploying them in some of the harshest environments on the planet? In the case of marine renewable devices, how should survivability against a 100 year wave event be tested, when by definition the likelihood of one occurring during testing is extremely small?

Many of the answers can be found in small scale experimentation and increasingly so on computer simulation using CFD techniques. However simulation of free surface flows for segregated multi-fluid problems is far from perfected. One of the principle areas of difficulty in simulating this class of problem is the accurate tracking of the interface separating the fluids being simulated, commonly the water-air interface in marine flow problems. Representing a free surface accurately as a continuous, smooth entity using a computer is a challenge that along with a great number of others has yet to be fully mastered such that real-world accurate results can be produced in a timely manner.

The principle area of research of this PhD project is on the numerical aspects of CFD, the underlying algorithms that make simulation of complex physical interactions possible. Because of the complex nature of fluid physics, generalised fluid flow problems are insoluble by analytical means, with only the simplest of problems yielding close-form mathematical solutions. Instead approximate numerical solutions are generated using a large numbers of calculations, ideally suited for powerful modern computers. Many of the techniques developed use a discrete numerical grid structure to represent the continuous fluid volume.

Structured numerical grids have been used extensively to solve CFD problems. In 2D, grids of regular quadrilaterals are commonly used as a basis for various CFD methods. However regular, structured grids reduce the complexity of geometries that can be represented, limiting in applicability for problems with complex geometries. The introduction of unstructured grids can help to improve the capability to represent complex geometry but not all CFD algorithms are suitable for solution using these. The particular focus of this PhD project is the development of a new method of maintaining an accurate free surface which is suitable for use on both structured and unstructured grids. Over the course of this chapter the motivation, pros and cons of both experiments and simulation are elaborated on. In Chapter 2 a more detailed description of the development and approach of the methods relevant to this thesis are presented as a literature survey.

1.1 Computational Fluid Dynamics

The field of theoretical fluid dynamics saw major development in the 18th and 19th Centuries with the first notable attempt to publish the fundamental equations that govern fluids by Leonard Euler 1757. The Euler equations describe fluid flow, neglecting fluid viscosity. The term applied to flow regimes by these equations is 'inviscid'. The lack of fluid viscosity dictates that these flows have no resistance to internal shear stress and do not develop any turbulence.

A further significant step was the introduction of the Navier-Stokes equation, which added viscosity terms to the Euler equation. The equation is named after Claude-Louis Navier and George Gabriel Stokes and was developed in the first half of the 19th Century. The equation is included as Equation 1.1 purely for the reader's interest.

$$\rho\left(\frac{\partial \mathbf{v}}{\partial t}+\mathbf{v}\cdot\nabla\mathbf{v}\right)=-\nabla p+\mu\nabla^2\mathbf{v}+(1/3\mu+\mu^v)\nabla(\nabla\cdot\mathbf{v})+\mathbf{f} \tag{1.1}$$

Both the Euler and Navier-Stokes equations are non-linear partial differential equations (PDEs) with no known general mathematical analytical solution. There are a some fluid flow problems that can be simplified to a linear form that is possible to solve analytically. Examples of such simplification are Stokes flow or Couette flow. However, these flow regimes are limited in their applicability and do not provide a general solution for complex fluid flow.

Because of the full Navier-Stokes equation's general insoluble nature, a variety of numerical methods have been developed to approximate the various quantities that appear in Equation 1.1. Through repeated calculation of approximate solutions on a discrete numerical grid, both steady state and transient flow problems can be simulated. High performance computers are ideally suited to the large volume of repetitive calculations required to do so. The fundamental quantities of fluid dynamics are mass, momentum and energy and great attempts are made to conserve them when simulating a problem. For both the Navier-Stokes and the Euler equations the principle solution is of the velocity field for the problem, with additional quantities derived from this solution for the most part.

Mentioned in the previous section, flow solutions can be calculated on a discrete numerical grid. The process of decomposition of the continuous real-world domain into a mathematically discrete world, suitable for solution on a computer is called discretisation. There are several approaches to domain discretisation, of which the two most commonly encountered and relevant to this Thesis are described.

The first method, which is the approach used exclusively in this Thesis, is that of Eulerian discretisation. The basic principle of Eulerian methods is that the continuous volume which is to be occupied by fluid for the duration of the simulation is decomposed into a grid structure. An example of a graded Cartesian grid can be seen in Figure 1.1 representing a multi-element

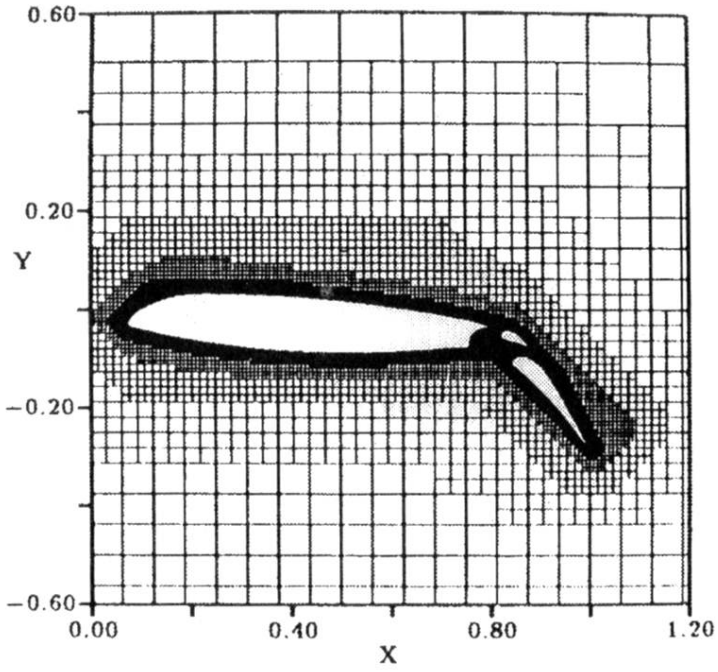


Figure 1.1: An Eulerian numerical grid representing the fluid volume around a multi-element aerofoil. Taken from Anderson (1995, 214)

aerofoil of an aircraft. The individual squares of the grid are commonly referred to as cells and the intersection of the grid lines represent points which may be used to hold the data of the CFD simulation. During the solution of the problem, the numerical grid remains stationary with respect to the fluid being simulated on it.

The second approach is that of decomposing the fluid itself into either discrete cell volumes or particles. The discrete volumes of fluid are tracked in their evolution through time. For methods using a grid of cell volumes, as the grid moves with the fluid it deforms over time. Care must be taken that the grid does not become 'tangled' or of insufficient quality to produce an accurate solution on. However one of the advantages of this method is it guarantees conservation of fluid mass throughout the simulation. Methods by which the frame of reference moves with the entity are commonly called Lagrangian methods, named after Joseph-Louis Lagrange whose doctoral supervisor was the famous mathematician and namesake of the first discretisation method, Leonard Euler.

While modern CFD methods can offer a huge amount of detailed information in the correct situation, there are still compromises that must be made to reduce problems to a complexity that is solvable in practical time-scales on modern computers. In the areas of research that have received the most attention such as supersonic flow regimes and detonations, simulations have become extremely accurate. The areas of turbulent, incompressible flow and free surface flow are still developing and while good results can be obtained, there are complex regimes such as

turbulent flows with a free surface that are difficult to resolve successfully.

Ultimately for the short to medium term future, and probably beyond, experimental and simulated data will both play an important role in the design of 21st Century technology. As Roe (1986) remarked, “a computer simulation does not have quite the same status as an experimental result at present as there usually remains some doubt about its accuracy ”, and while the margin has closed in the 25 years since it was written it still remains true today. In the following sections some of the advantages and disadvantages of conducting experiments and simulations are discussed.

1.2 The Strengths and Weaknesses of Experiments

Experiments have been used extensively to aid the design of new structures and machines through the discovery of, and the better understanding of physics. Until the invention of the digital computer, approaches to engineering design were limited to analytical mathematical calculation and the gathering of data through experimentation where the physics governing problems was so complex it was not possible to resolve to a sufficient degree of reliability. Experimentation can be of vital importance in cases where high levels of assurance were required such as the design of aircraft. However it is also desirable to minimise the requirement for experiments as they generally tend to be a time consuming and expensive process.

One of the immediate advantages of fluid flow experiments is that in the majority of cases they can be conducted using the same fluids and materials that are used in the real-world application. Complex geometries and devices can be reproduced precisely to scale, or full scale where appropriate, with moving parts included. This immediately promotes a level of confidence that the constituents of the experiment behave like their real-world design counterparts.

Because the laws of physics apply to the real-world, all physical effects are present and act all the time. Computer modelling is still at a stage such that simplification is required, only including the most important physics phenomena to the situation. Sometimes simulated physics is only a good approximation of the real-world physics in certain parts of the simulated problem. Neither of these issues effect experiments.

Experiments can range from being simple and inexpensive to prohibitively complex and expensive. In the case of experiments relating to marine flows, water is an essential requirement necessitating the use of potentially expensive equipment such as high speed, high resolution cameras or particle image velocimetry (PIV) equipment. Large pieces of apparatus such as a wave tank may be required. One such facility is The University of Edinburgh's Curved Wave Tank (Taylor *et al.*, 2003). It is capable of generating fully realistic 3D sea states in an 8 meter wide, 1 meter deep tank, using 48 wave makers.

The Curved Wave Tank requires a large laboratory to house it and trained staff that are experts in its operation. It took 10 months to build with a budget that could only be considered feasible for the creation of a long term experimental facility. While the commitment to its construction, upkeep and operation is large, it is capable of creating precise waveforms at a convenient scale and allows the creation of rare sea states to be repeated consistently in a warm, safe environment.

With the obvious complexity of these experiments comes an expense of time. Aside from the time scale required to design manufacture and install experimental equipment, there is a practical limit to the number of experiments that can be run for different cases. The man power, set-up time and processing of results makes it difficult to achieve the large numbers of experimental runs that may be required for an optimisation project. Therefore the value of experimental data

can be extremely high and care must be taken to record all the potentially important data. When conducting experiments to better understand a complex problem, determining what data should be obtained can be a significant challenge.

Another difficulty is the management of measurement errors. High measurement accuracy is very possible to achieve in well designed experiments but the potential for poor result taking is always present. In most cases, measurement error can be anticipated and minimised in the initial stages of experiment design, however to do so relies heavily on precise understanding of the experiment and the experience to identify issues for consideration. If important aspects of an experiment are overlooked then the potential for extensive redesign exists, costing more time and expense.

A significant limitation of fluid flow experiments is that their dynamics do not scale with the experiment geometry. For a detailed explanation Douglas *et al.* (2001)[p299-304] provide an excellent source. The principle dimensionless ratios of fluid flow are Froude, Mach, Reynolds and Weber numbers. When scaling an experiment these numbers do not necessarily remain constant. As an example, of importance when dealing with free surface flows is the Weber number. It is a ratio of a fluid's inertia compared to its surface tension. As scaling occurs the Weber number changes with the fluid inertia value decreasing while surface tension remains constant, causing problems when studying breaking wave behaviour. By way of a further example, it is impossible to maintain both a constant Froude number while also maintaining the Reynolds number. This is a particular problem when experimentally modelling tidal turbines as turbulence and free surface effects cannot be maintained simultaneously.

An important and successfully studied area of research, relevant to the interface tracking method featured in this PhD project is that of wave over-topping of seawalls. As Hu *et al.* (2000) writes, a large amount of damage and flooding can be caused by wave strikes whose volume is not retained by sea wall defences. The Violent Overtopping of Waves at Seawalls (VOWS) project (Allsop *et al.*, 2011) was conducted as a partnership between Edinburgh, Sheffield and Manchester Metropolitan Universities, as well as non-academic institutes including HR Wallingford. The project combined a series of investigation techniques including 2D experimental results using the wave flume at Edinburgh University, 3D experimental results from the wave basins at HR Wallingford and a computer modelling study in the form of a 2D wave flume code, Amazon-SC, and a 2D wave basin code, Amazon-CC.

While the challenges of successful experimentation are daunting, in large areas of research in science there is simply no substitute. The advantages of working with real materials and under full physical effects mitigates the obstacles that must be overcome. Experimental data can be extremely useful, accurate and delivered with a high degree of certainty. However the current state of computer simulation, in general, has a great deal of value to add to the experimental data that science has traditionally relied on.

1.3 The Advance of Modelling Fluids on Computers and the Current Limitations

The development of physics simulation on computers is a comparatively new field, having existed for approximately 60 years. In that time development has been rapid and the power of modern computers currently allows for simulation of some highly complex problems. A key area of physical modelling has been Computational Fluid Dynamics, a key tool in the advance of engineering design, in the field of fluid flow. While CFD simulation data has not replaced experimental data it has quickly become an essential complementary partner to it, helping engineers design more sophisticated devices quicker and for reduced costs.

In the previous section it was highlighted that fluid flow experiments can be time consuming and expensive. In some cases flow regimes can be impossible to recreate in a laboratory, such as supersonic flow past a jet. In the case of breaking waves on a beach or against a sea defence, the complex velocity field in parts of the flow is impossible to measure with any meaningful accuracy. Computer modelling of fluid flow can avoid many of the disadvantages encountered when undertaking experiments and it can also provide an opportunity to obtain more detailed information about the flow fields. Velocity vector fields and pressure contours can be displayed over the computational domain through post-processing the simulation data. The power to manipulate such detailed simulation data means that peak pressures, locations of high velocity, maximum turbulence, flow separation points are all possible to determine and easily accessible for analysis. Further characteristics, such as drag, can be derived from the primitive variables of the simulation and the opportunity to re-run the simulation at a later date is possible.

One of the most effective manners with which simulation data can be used is in the practice of experimental validation. The accuracy of CFD simulation data can be validated against small scale experimental data. Once the validity of the physics of the simulation has been established, the confidence in the results from full scale simulations is greatly increased, without the need for creating a full scale experiment. The potential time and cost savings of simulating a full scale prototype without the need for manufacturing it are obvious.

While the benefits of CFD simulation are clear there are still a great deal of issues that demand continued research and development. Computing hardware has increased in power at an exponential rate lowering the relative costs (Figure 1.2) however engineers are still limited in the scope of CFD problems by the power of their hardware. Hirsch (2007, p6) outlines how those limits changed in the last 30 years of the 20^{th} Century. For practical purposes the computer solutions of 1970s were limited to potential flow and 2D inviscid flow problems. Low speed processors took large amounts of time to compute answers and the memory used to do so was limited in size and expensive. In modern marine flow problems, heat transfer effects are regularly neglected as they play little part in the behaviour of the fluids in this regime. Fluid turbulence is also simplified and approximated to reduce the computational requirements.

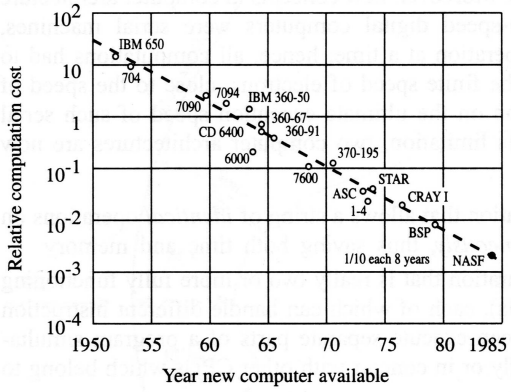


Figure 1.2: The relative price of computation vs time. Taken from Anderson (1995, p27)

Development in computers to the present day has resulted in desktop PCs that are capable of such a large number of calculations - 50 – 100 GFLOPS (Giga-Floating point Operations Per Second) is now the normal range for a high performance desktop PC. This has created the possibility to simulate complex, multi-dimensional problems with relative ease. However the raw calculation power of the computer does not solve many of the problems that are still encountered when trying to accurately model flow physics using discrete numerical approximations.

While computational constraints such as memory limits and instructions per cycle are less of a restriction to the computational fluid dynamicist, the fact remains that numerical schemes must be simple enough for the programmer to implement efficiently. Indeed, with CPU clock speeds having reached a practical ceiling around 3.5GHz, the trend is for parallelisation of simulations across multiple CPU cores or even computers. To do so requires algorithms that are suited to such techniques and allow for concurrent operation on data structures. It is now common for shared memory computers to be capable of running 12 software threads simultaneously, maximising possible computation power. It is extremely challenging to design mathematical algorithms and write computer code for complex numerical schemes that utilises those threads efficiently.

The emerging technology, set to increase the power available to CFD even further is GPGPU (General Purpose Graphics Processing Unit). Because of a GPU's SIMD (Single Instruction Multiple Data) architecture it is capable of a huge number of FLOPS allowing for the opportunity to run simple fluid simulations in real-time, such as the method presented in Fernando (2004, Chaper 38). Segregated two-phase flows are already successfully being simulated on GPUs (Zaspel and Griebel, 2010).

The creation of numerical meshes is another fundamental area of CFD that receives continued research. Robust, adaptive meshes that maximise computational efficiency while maintain accuracy are a challenge to the development of numerical schemes. This is especially true in transient simulations with moving boundaries where they are required to be re-meshed as

boundaries move. The resolution of the mesh has a direct effect on the accuracy of the numerical solution but also an impact on the computation time. A good mesh can have areas of high resolution cells where the solution undergoes rapid variation and is of coarse resolution where the solution remains nearly constant.

While structured Cartesian grids are both straightforward to create and design numerical algorithms for, unstructured grids have an advantage in their ability to conform to more complex geometries and are capable of undergoing deformation during problem solution to account moving geometries. In areas of small displacement, it is possible to modify the cell structure without the need for the complete re-meshing of the problem domain. However the shape of unstructured grid cells can have an strong effect on the accuracy of solutions and effective numerical algorithms are more difficult to design and implement. Therefore an effort to develop hybrid grid methods such as the Cartesian cut-cell grid method have been ongoing. These grids use a majority of Cartesian cells with some 'cut-cells' to account for geometry dimensions. Their ability to adaptively refine the mesh in areas of importance, while still maintaining a good quality cell shape, is highly attractive. However the cut-cells produce a variety of irregular polygons and therefore any fluid flow solving algorithms used on them must be specifically developed to manage their mixed geometry mesh.

When examining current research in marine engineering there is a continued need for the development of free-surface flow solvers. Watanabe and Saeki (2002); Watanabe *et al.* (2005) developed free surface simulations modelling the break up of rolling waves and the velocity field of their surf zones. In related work, Saruwatari *et al.* (2009) developed simulations of the break-up of waves and their finger jets. The study of phenomena in the break-up zone of waves is important to understand not only for the study of wave impact on sea wall defences but also for the influence of the atmospheric environment and beach erosion. Gao *et al.* (2006); Ingram *et al.* (2009) developed a successful CFD code that is capable of simulating breaking waves over-topping sea defences. The work takes advantage of Cartesian cut-cell grids and is capable of representing both angled dikes and vertical sea walls.

There is already a significant body of successful research benefiting from free surface flow solver technology. Climate change, flooding, sea defences and beach erosion are already being studied with some success. However there is continued research required to create more accurate, more efficient free surface flow solvers that can deliver the research requirements of marine renewable energy sector. Improved interface tracking using a level set method on a Cartesian cut-cell or fully unstructured grid is one of the ways in which free surface flow solvers can be improved, offering a more realistic representation of the free surface on a robust, efficient numerical grid.

1.4 An Outline of the Thesis

As previously stated, the particular focus of this PhD project is the development of a new method of maintaining an accurate free surface that is suitable for use on both structured and unstructured grids. To accomplish this objective a level set method is implemented as a interface tracking method. To improve the accuracy of the level set method, a reinitialisation routine is used between simulation time-steps. To successfully implement the reinitialisation routine on a unstructured or hybrid grid, such as a Cartesian cut-cell grid, modifications must be made to the gradient approximation of the reinitialisation routine. It is this work that forms the majority of the novel contribution to science in this PhD project.

This Thesis is divided into a further six chapters. In Chapter 2 the reader is presented with a review of literature that was considered before and during the PhD project. This literature review is reasonably wide ranging in subject matter and is designed to give the reader a picture of the scientific methods relevant to the PhD project, before a list of specific project aims and objectives is presented.

Before the merits of an interface tracking method can be assessed, some method for moving that interface must be implemented. In Chapter 3 an advection solver called the CIP method is introduced, implemented and tested in the 1 and 2D. To perform the testing a protocol known as a grid convergence study is described and used for both 1 and 2D cases. A short test example of interface tracking is presented and the motivation for improving this discussed.

Chapter 4 describes the popular interface tracking method known as the level set method which forms the focus of the novel contribution of this PhD project. The theory of the level set method is introduced before a method for improving its accuracy is described. The method is known as level set reinitialisation and theory and test cases in both 1 and 2D are covered before extending this theory to an improved form with an additional area preserving constraint.

In Chapter 5 a novel method of performing the level set reinitialisation is presented. This method is new because it has the potential to be extended to unstructured grids of an arbitrary geometry cell shape. The method is developed specifically with the Cartesian cut-cell numerical grid approach in mind. However the method has the potential to be extended to any unstructured grid.

Finally, in Chapter 6 all of the level set reinitialisation methods are tested and compared thoroughly in a side by side comparison, with a short optimisation study for the best performing method. The conclusions and the continuation of future work are summarised in a short chapter, Chapter 7 at the end of the Thesis.

All of the numerical methods in this thesis were implemented from scratch by the author using FORTRAN 90. The results published throughout this thesis are generated using data output from the author's code library unless otherwise stated. A short comment on the programming experience that was gained during the project is included in Appendix A.

Chapter 2

Literature Survey

Computational physics, of which computational fluid dynamics is a part, is a vast and interesting subject area. It would not be possible to attempt to cover the full range of methods that could be used to simulate a breaking ocean wave in the scope of a PhD project. For this reason a few key areas of research are covered in this literature review. The principle areas of review consist of numerical grid techniques, Eulerian interface tracking methods and high order accuracy gradient schemes. There is also an aside included describing some basics of finite difference techniques and the fundamental difficulties faced when implementing them. This is included for the less knowledgeable so that they may better understand the motivation and terminology behind gradient approximation methods.

Should the reader wish to gain a greater perspective of computational fluid dynamics, the development of the field and description of range applications and methods, the books by Anderson (1995) and Hirsch (2007) are highly recommended and have helped enormously to frame this PhD project's relevance in the wider field. Further to these books, should the reader wish to gain in depth knowledge of the level set method, on which this thesis focuses, the published works of Sethian (2001) and Osher and Fedkiw (2002) are essential reading. "Level Set Methods and Dynamic Implicit Surfaces" (Osher and Fedkiw, 2002) is particularly well suited to the novice, guiding them at a high level through a wide range of numerical techniques and applications for level set methods.

The sections of this literature survey are written in more or less chronological order, setting out the development path of much of the relevant work. There is little technical detail or any equations included, the relevant information is contained within the main chapters 3-5. However the reader should gain an understanding of why a number of the decisions were taken to develop an interface tracking method for Cartesian cut cell grids based on a novel level set reinitialisation scheme. At the end of this chapter a specific description of the aims and objectives of this project are described before moving to the core of the thesis.

2.1 Numerical Grids

In the field of computational fluid dynamics there have been a great number of techniques and methods developed for simulating fluid flows. A large number of these methods, for a range of flow regimes, have been implemented using an Eulerian grid. The numerical grids upon which so much of CFD is based are fundamental to the process of discrete mathematical methods, the accuracy and the stability of the solution. Grid generation can be extremely complex to develop when dealing with methods for generalised geometries. The production of good quality numerical grids can take a significant portion of a CFD project's time. The selection of the correct grid type and CFD methods for a project is of paramount importance if meaningful results are to be obtained.

A numerical grid is used to store the relevant simulation data at known points in the volume occupied by fluid. Regardless of the mathematical discretisation technique used, a grid is a distribution of points chosen throughout a simulation domain. These points can be thought of as the vertices of cells that describe the grid structure such as the example in Figure 1.1. There are many ways of choosing the points in a grid, which can lead to a number of different types of grids, for example unstructured or adaptive grids. The mathematics behind grid generation is extremely complex and out-with the scope of this PhD project. Therefore this section is a high level overview of some of the grid methods that are commonly used to form the basis of CFD simulations.

The numerical grids introduced here fall broadly into two categories, structured and unstructured. The difference between theses grid types is straightforward. Structured grids place their points in a geometrically predicable manner, aligning the point distribution either with the principal axes of the problem or around the geometry of the solution volume's boundaries. Unstructured grids have an altogether more arbitrary distribution of points, however to maintain solution accuracy, even these grid types grade their cell sizes in a progressive manner that resolves the flow field in an appropriate manner. An alternative definition presented in Hirsch (2007, p250) is, *"structured grids are composed of families of intersection lines, one for each spatial dimension, where each mesh point is located at the intersection of one line, and only one line, of each family."* Whereas unstructured grids, *"refer to arbitrary distributions of mesh points, where the points are connected by triangles, quadrilaterals or polygons."*

2.1.1 Structured Cartesian Grids

Uniform Cartesian grids are straight forward for performing mathematical calculation and therefore represent the ideal solution accuracy, being applied whenever possible. As described in Osher and Fedkiw (2002, p6), the most popular grids, are Cartesian grids. In a uniform Cartesian grid, all the subintervals are equal in size, and the spacing $\Delta x = x_{j+1} - x_j$ is consistent over the whole grid. Furthermore, it is usually convenient to choose $\Delta x = \Delta y$ so that the approximation errors are the same in the x-direction as they are in the y-direction. The advantage of this is that structured grids are favourable in accuracy, CPU efficiency and memory requirement at the expense their flexibility to the types of geometry they are capable of representing.

As an extension of uniform Cartesian grids, non-uniform grids vary their resolution in each dimension throughout the simulation domain to improve the representation of quantities in specific zones. A common application of this is the refinement of the perpendicular grid axis close to a boundary as this allows for the boundary layer to be highly resolved. Graded grids require little increase in mathematical complexity and do not obstruct the application of many finite difference schemes. However, they are best suited to steady state simulation, where the important flow features have no ability to shift in location.

A challenge presented by many simulations, is the ability to include adaptive areas of high resolution mesh in zones of significant flow features on a coarser background mesh. Utilisation of a fine mesh over the entire domain is not an efficient use of computational effort. While it is possible to construct a straightforward Cartesian mesh with intelligent use of grading in steady state flow problems, this is not possible in transient simulations where the flow characteristics are likely to be changing.

A popular solution to this is adaptive mesh refinement (AMR) with one of the first methods proposed by Berger and Oliger (1984). Using a base level coarse mesh grid, a simulation is started. As the simulation progresses, an estimation of the error on each cell is produced. If the error estimate exceeds its tolerance level the grid inside that zone is refined. In 2D the grid cell is divided in half in each dimension, producing four cells at the finer grid level, in 3D this produces eight new sub-cells. The use of quadtree (2D) and octree (3D) data structures in computing, summarized well by Losasso *et al.* (2006), can be implemented to produce a highly efficient structure to which cells can be rapidly created and deleted. Therefore the cost of maintaining the method is far less than the cost of any additional grid points in unwanted areas of the domain.

AMR has been used extremely successfully for a range of applications such as high resolution shock capturing in super-sonic flow problems (Berger and Colella, 1989). The areas of highest grid refinement tend to occur around the shock interfaces, allowing for a sharp representation of the discontinuous shock wave. Coirier and Powell (1995) used AMR to assess the accuracy of Cartesian grids, representing two aerofoils in 2D. The mesh refinement makes it possible

not only to capture the important flow features between the aerofoils but also refine the grid around the geometry boundaries, helping to produce an accurate representation of the real world geometry. Nourgaliev *et al.* (2005b) demonstrated improved interface tracking accuracy using the level set method with AMR by refining the resolution of the grid about the interface position. This represents a greater challenge, because of the transient nature of the simulation, the adaptive grid must be updated at every time-step for the high resolution grid to track the interface.

2.1.2 Boundary Fitted Structured Grids

When curved solid boundaries are present in a simulation they cannot form part of the Cartesian grid. Within the bounds of structured grid methods two options exist, either the Cartesian structure is kept, relying on grid refinement to adequately resolve the curved surface (Coirier and Powell, 1995), or a boundary conforming grid is used. Curvilinear grids are a 'body fitted' grid type that operate on using a spatial transformation. Using such a transformation, it is possible to compute a non-uniform grid structure as a uniform grid in computational space. Anderson (1995, p192-200) provides a comprehensive description of the mathematics.

This type of grid has been widely used in the generation of grids for aerofoil sections. The transformation of these types of grid can be treated as boundary value problem to which a solution can be sought through the solution of an elliptical partial differential equation. Kreiss (1983) presents a straightforward manner with which to construct a grid in the area between two boundaries using splines and later solving a hyperbolic equation to complete the grid transformation.

The work of Koshizuka *et al.* (1990) produced grids for the simulation of incompressible flow using the SMAC algorithm (Amsden and Harlow, 1970). In their example problems, the simulations performed on regular Cartesian grids are compared to those on a boundary fitted grid. The regular Cartesian grid is incapable of representing the angled boundaries of the test problems, using a discontinuous stepped approximation, where the boundary fitted grid succeeds. The differences in the flow profile close to the boundaries on the two grids are significant, with the irregular boundary disturbing the true flow path of the fluid. Such an example shows the importance of the ability to accurately represent complex boundaries in flow problems.

2.1.3 Unstructured Grids

While the grid methods that have been discussed so far, may not be uniform, they follow a structure of some kind. With the introduction of finite volume CFD algorithms, it has been possible to move to a newer class of grids that follow little or no structure, the unstructured grids. The introduction of arbitrarily unstructured grids has allowed the inclusion of more complex boundary geometries and the possibility of moving boundaries with the mesh deforming at

each time-step. Some of the earliest unstructured grids have been widely utilized to model the complex geometry around aerofoils (Mavriplis, 1988) and other aerodynamic problems (Frink *et al.*, 1995).

Despite the high degree of accuracy offered by structured grids and their computational efficiency, unstructured grid methods are preferred by commercial industry (Venkatakrishnan, 1995a). The ease of mesh generation for complex geometries and algorithms that allow for automatic grid generation are important advantages offered by unstructured grids. The use of triangular in 2D and tetrahedral grid elements in 3D is common, using techniques such as Dirichlet tessellation for automatic grid generation (Bowyer, 1981). Triangular grids have the advantage over AMR meshes of allowing for the addition of extra nodes during grid refinement without leaving hanging nodes.

With respect to segregated flows, Barth and Sethian (1998) point out that in moving interface problems, jump conditions across the boundary are critical to both solutions of the partial differential equations on either side of the interface. Interpolation of terms to neighbouring grid elements can be delicate but an interface-fitted coordinate system offers a straightforward way to build these terms. As such, some researchers have taken advantage of the unstructured grid's ability to conform accurately to smooth boundaries. Unverdi and Tryggvason (1992) explicitly represented the gas-liquid interface of a rising bubble using unstructured grid elements. However, methods that require this type of grid deformation rely on a somewhat steady and predictable propagating interface to prevent the interface mesh from distorting grossly between time-steps.

Allowing a numerical grid to distort due to boundary movement can lead to high aspect ratio or 'skewed' cells. This contributes to grid errors and ultimately highly distorted grids requiring re-meshing to maintain acceptable results. Authors such as Murayama *et al.* (2002) have introduced straightforward methodologies such as the use of spring coefficients to allow boundary nodes to move with moving boundaries and 'stretch' the connecting grid edges. However there is still the risk that large displacements cause grid cell entanglement for which the only solution is to re-mesh. Re-meshing at any time is undesirable, the required interpolation causes errors in the terms of the equations, leading to inaccuracy and ultimately a loss of physical conservation (Lohner *et al.*, 1999). The computational expense of re-meshing can also be prohibitive, if performed regularly. To improve this, Moyle and Ventikos (2008) suggested a novel method for local re-meshing in areas of high amplitude displacement.

2.1.4 Cartesian Cut-cell Grids

Cartesian cut-cell grids are a more recent method for creating a numerical grid whose points have the ability to conform to boundary geometries of arbitrary complexity while the principle structure is that of a regular Cartesian grid. Ingram *et al.* (2003) write that the method was originally developed for potential flow problems before being extended for use with the Euler equations. Due to their computational efficiency in comparison to other unstructured grids, Cartesian cut-cell grids have been used to simulate problems with both static and moving geometries in Yang *et al.* (1997a) and Yang *et al.* (1997b).

As the name suggests, a Cartesian cut cell grid can be generated by cutting a solid boundary from a background Cartesian mesh. The result is a hybrid grid composed of structured and unstructured grid cells. The great advantage of Cartesian cut-cell grids over triangular element based unstructured grids is, for a given geometry, only a small number of cells of the background mesh need to be altered in order to produce a geometry conforming grid. This improves computational efficiency when compared to other unstructured methods and becomes a huge advantage with the inclusion of a moving boundary such as in Causon *et al.* (2001). Rather than requiring the full domain to be adjusted or re-meshed to accommodate geometry changes, a small number of cells must be 're-cut' to reflect the change. Yang *et al.* (2000) provides a detailed description of a moving boundary cut cell method. Because the vast majority of the numerical grid consists of regular Cartesian cells, solution accuracy can be maintained to its highest level. There is no danger of highly skewed grids influencing the quality of solutions. Furthermore the method can be successfully integrated with adaptive mesh refinement techniques (AMR) to produce an extremely versatile and computationally effective numerical grid, as demonstrated in a 2D method by De Zeeuw and Powell (1993).

In the fields more closely related to marine engineering, Causon *et al.* (2000) presented a Cartesian cut-cell method for the shallow water equations enabling a boundary conforming mesh for complex geometries. Their quadtree cut cell mesh was used to resolve the coastline and waterways of the Netherlands with high accuracy. Cartesian cut-cell grids have also been used in multi-fluid simulations. The authors Qian *et al.* (2003) implemented a two fluid, incompressible Euler equation solver capable of simulating free surface flow and presented waves striking a sloping beach. The method was further developed to solve the full Navier-Stokes equations by Gao *et al.* (2006) and used to demonstrate wave run-up on a smooth sea dike. The Cartesian cut cell grid was capable of representing the sloped beach in an otherwise regular Cartesian grid. Finally, Ingram *et al.* (2009) presented work simulating violent wave over-topping of sea walls, using the results to predict the volume of over-topping discharge. Again, in this example the cut cell grid allowed straight forward representation of a more complex geometry in a surrounding uniform Cartesian grid.

2.2 The Development of Eulerian Interface Tracking Methods

Interface tracking methods have been implemented in CFD for nearly 55 years (Harlow, 1957). In their infancy, computer power and memory was limited such that only a single liquid phase could be simulated in 2D (the gas phase, air, was neglected in the simulation). Today, due to the increase in computer power and the volume of research, it is possible to simulate free surface flow with turbulence and surface tension effects in 3D (Saruwatari *et al.*, 2009) for short periods of time.

In this section, a review of the most popular Eulerian interface tracking methods is presented and split into four separate categories. The first category includes methods that use a Lagrangian marker particle to seed the liquid fraction of the simulation and are amongst some of the earliest methods to track two fluid flows. Volume methods form the second category. These methods use a fraction of fluid in a cell approach to determine which cells contain the fluid interface. While these methods have seen widespread adoption, difficulties in calculating a realistic interface detract from their successes. Implicit tracking methods have seen a great deal of development in more recent years. This class includes the level set method which is the subject of the third subsection. The relatively computationally cheap cost and ability to represent smooth, continuous interfaces offer advantages over the previous two approaches. However level set methods present their own challenges and maintaining the accuracy and subsequent mass conservation requires effort. Finally some hybrid methods that combine the benefits of marker method or volume methods with the implicit level set scheme are covered as the fourth category.

2.2.1 Marker and Cell Methods

The Marker and Cell (MAC) method is one of the earliest two fluid CFD methods, enjoying strong popularity through the 60's and 70's. Its origins can be traced as early as 1957 from the publication, Harlow (1957). However it was in the publications Harlow and Welch (1965a,b) that the principles of the method came to be known as the Marker and Cell Method. In Harlow *et al.* (1965) the method was successfully demonstrated simulating a 2D liquid wave.

As the name suggests, the method uses Lagrangian marker particles in Eulerian grid cells as a means of tracking fluid flow. MAC methods offer the advantages brought by both Eulerian and Lagrangian reference frames as the marker particles prevent mass loss and diffusion of the liquid, helping maintain a defined interface. The diffusion prone Eulerian grid improves flow tracking and stability of rapidly deforming fluids. The interface is represented by the partially full cells or full cells adjacent to empty cells. The particles of the MAC method are used as a fluid marker only and do not interact with each other or the flow. This method is appropriate for segregated fluid flows between a fluid and gas phase where only the flow physics of the liquid is resolved.

Amsden and Harlow (1970) suggested a simplified method of MAC, calling it SMAC. In the method the boundary conditions and pressure calculations were simplified but ultimately one of the most costly aspects of the method was the calculation of the marker particle positions. While the scientific research computers of the 70's were sufficiently powerful to solve 2D problems, the increased cost of 3D simulation was significant due to the increased numbers of particles required. A further obstacle to the method is the bunching of particles, due to local flow velocity, requiring their periodic redistribution while preserving simulation accuracy.

Another consideration of MAC methods is the lack of a free surface, it must be explicitly calculated based on the distribution of particles. This adds an additional phase of computation to the method and one that is by no means straightforward to complete. Surface boundary conditions are difficult to implement and as a result potentially inaccurate. Nichols and Hirt (1971) produced a set of free surface boundary conditions that took into account both normal and tangential surface stress, producing a smoother, more natural surface to simulations. However, there are complications during the break-up and coalescence of the surface sections. This requires further deletion and redistribution of the marker particles and can lead to complex wave simulation becoming unmanageable.

While MAC methods offer a successful manner by which to track liquid phases in segregated flow problems, the computational costs of using Lagrangian marker particles in 3D is high. The requirement of explicit reconstruction of the fluid interface presents further complexity. This lead to the decline in popularity of the method by the 80's as fluid volume methods provided a simplified alternative method.

2.2.2 Volume Tracking Methods

Volume of fluid (VOF) methods were introduced in the 1970s (Rider and Kothe, 1998) and essentially represent segregated fluids in a similar manner as MAC by explicitly marking how much fluid is present in a given grid cell. While the MAC method does this using multiple particles held in grid cells, VOF methods use a single volume fraction marker ranging between 0 and 1. This approach helps to increase the computational efficiency and its extension from 2D simulation to 3D is trivial. The method has been sufficiently successful to be widely adopted by commercial CFD codes such as FLOW-3D, Ansys CFX and Ansys Fluent for multi-fluid simulation.

According to Scardovelli and Zaleski (1999) the strengths of modern VOF methods are their ability to preserve mass in a natural way with break-up and coalescence of fluid being handled automatically as the topology is implicit in the algorithm. Grid cells with a fractional, or mixed, fluid volume contain the fluid interface. However further calculation is required to determine the position and orientation of that interface. Without a surface reconstruction method, volume of fluid methods lose the sharpness of a fluid interface and smear the surface over a band of grid cells. This is undesirable due to the non-physical representation of the free surface.

The mass conservation properties of VOF methods can be problematic when numerical truncation errors create volume fractions of greater than 1 or less than 0. This can create small fragments of 'flotsam and jetsam' as described by Noh and Woodward (1976). Mass conservation leads to ejection of small non-physical packets of fluid leaving the main flow regime. The issue of flotsam and jetsam becomes particularly important when using a 1^{st} order interface method such as SLIC. As Scardovelli and Zaleski (1999) comment, the method requires the introduction of surface tension to prevent the interface becoming unstable and progressively destroyed.

The simple line calculation method (SLIC) is a robust VOF method developed by Noh and Woodward (1976). The method uses a simple interface who's position is perpendicular to the axis of propagation, creating a 'blocky' representation of the interface. The primary reason for this approach is the relatively simple algorithm that the method uses has the ability to represent more than two different fluids in each grid cell. The SOLA-VOF method developed by (Hirt and Nichols, 1981) uses a similar interface reconstruction with the additional improvement of allowing the interface to align with multiple grid axes, creating a 'stair-stepped' interface representation.

The original VOF methods of SLIC by Noh and Woodward (1976) and SOLA-VOF by Hirt and Nichols (1981) worked well in areas of small curvature but were only 1^{st} order accurate finite difference discretisations. The methods become inaccurate in areas of high curvature. A more accurate method of processing interface orientation is the FLAIR (Flux Line-Segment Model for Advection and Interface Reconstruction) method by Ashgriz and Poo (1991). FLAIR differs from previous methods by representing the interface as a set of line segments fitted at the boundary of two neighbouring cells. The surface orientation is entirely independent of the grid and continuous between the grid cells.

Other higher order methods that use PLIC (piecewise linear interface construction) methods such as Pilliod and Puckett (2004) reconstruct the interface with 2^{nd} order accuracy where the interface is smooth, however there is no requirement for continuity of the interface between cells, leading to small discontinuities on a cell-by-cell basis. In areas where the surface is discontinuous, such as points of coalescence, interface accuracy is poor. Furthermore, to maintain the volume conservation property of the method, the position is usually calculated iteratively, with adjustments made to the position of the interface, resulting in a computationally expensive process.

While VOF methods can be expensive and produce a non-physical, discontinuous piecewise interface representation, they are a successful and popular class of multi-fluid solver. VOF methods have been used to simulate breaking waves with Kleefsman and Veldman (2004) successfully simulating dam break and green water ship problems. However another method of Eulerian interface tracking, the Level Set Method, has grown in popularity over the last two decades and it is capable of smoother representations of a fluid interface.

2.2.3 Level Set Methods

The Level Set Method is an interface tracking technique introduced by Osher and Sethian (1988). Described as a signed distance function, the level set is a scalar function who's magnitude represents its distance from an iso-surface, in 3D, or iso-contour, in 2D. The sign of the function is determined by which side of the surface the level set occupies. Therefore a surface representing a front, interface or discontinuity of another kind can be tracked by finding the location at which the level set function is zero.

The advantages of using a signed distance function to track a surface are numerous. The level set can be stored as a single extra dimension on a numerical grid. It easily extends from 2D to higher dimensions, not something that is necessarily true for VOF or MAC. The interface is implicitly captured by the function allowing for a sharp, smooth and continuous representation. The propagation of the interface can be a function of its own geometry, in the case of curvature driven propagation, or advection by an external velocity field (Watanabe *et al.*, 2008).

Since its inception, the level set method has been used for a great number of applications. Osher and Sethian (1988) originally suggested it as a method for tracking flame front propagation and crystal growth. The method was then quickly adopted for use with the compressible fluid flow equations by Mulder and Osher (1992). The Hamilton-Jacobi level set formulation was incorporated into the compressible flow equations using two approaches. The level set function was solved in a non-conservative form using an external velocity, obtained from the standard conservative differencing of the hyperbolic system. This method was compared with the direct incorporation of the level set function, creating five conservative equations in 2D. Where density is discontinuous across an interface, with a smooth velocity function, the non-conservative level set method performed optimally while a fully conservative discretisation is better for a discontinuous velocity field.

In the paper by Unverdi and Tryggvason (1992) a bubble rising in fluid was simulated using an incompressible conservative scheme. In the work it is correctly stated that conservative numerical schemes are likely to suffer from excessive numerical diffusion at discontinuous areas of data where low order gradient approximations are used. Higher order gradient approximations suffer from numerical oscillations known as Gibbs Phenomenon (Section 2.3) at discontinuities leading to loss of accuracy, the possibility of non-physical results occurring and numerical instability. The solution presented by Unverdi and Tryggvason (1992) was to explicitly track the motion of the interface represented with a triangular grid. This allowed a different fluid and viscosity to be simply defined either side of the thin interface, avoiding the possible problems other numerical schemes encounter.

The disadvantage of explicit interface tracking is that it is difficult to implement, especially if the interface undergoes rapid deformation and possible breakup. The level set method is suggested as a simpler method by Sussman *et al.* (1994) for simulating a rising bubble. The

fluids used in the simulation were air and water with a density ratio of 1000:1, far greater than the 29:4 ratio simulated in Mulder and Osher (1992). However, as simulation time progresses, the velocity field moves the level set function and after some time it ceases to be a signed distance function. The loss of signed distance causes the gradient of the level set function to shift away from 1 or -1. In areas of extremely high or low gradient the accuracy to which the level set function can track the interface position is reduced. The result of this in the case of the bubble simulation is a mass loss of 41% as the level set fails to track the interface accurately.

The solution to the loss of the signed distance property was first suggested by Chopp (1993). A level set method was used to study minimal surfaces and it was found that a loss of the distance property of the level set function occurred when boundaries were applied to a level set curvature driven problem. The solution proposed was to re-distance or 'reinitialise' the level set function at regular time intervals during the simulation. The simulation was paused to calculate the position of the zero level set and then calculating the distance to it from each grid point. The reinitialisation idea was extended further in an implementation for incompressible two phase flow by Sussman *et al.* (1994) and later Chang *et al.* (1996). The method by Sussman *et al.* (1994) introduced a more sophisticated approach which did not require the zero level set to be found explicitly. An iterative equation was repeated until such time as it reached a steady state. The steady state solution was an excellent approximation to the signed distance function. The added advantage of this approach was that re-distancing started at the interface position and moved outward. This allows for the reinitialisation procedure to only be iterated until an adequately thick band was re-distanced about the zero level set, reducing computational effort. The reinitialisation method was further improved in accuracy by Sussman *et al.* (1998) using an improved Runge-Kutta time integration method and a higher order ENO gradient approximation scheme. Despite the attempt to improve the accuracy of the reinitialisation method the zero level set was found to be moved a small amount during the procedure, resulting in small amounts of mass loss proportional to the number of iterations that were conducted. A further improvement, to stop movement of the zero level set, was made in Sussman and Fatemi (1999). By attempting to preserve the mass in the immediate vicinity of the interface through application of an integration stencil, mass loss during reinitialisation was reduced to a minimum.

While the level set method proved to be a robust, efficient way to track interfaces, Sethian and Smereka (2003) explain that including the level set function in a problem is equivalent to adding an extra dimension to the problem. Given that the level set function is only required to be known in the vicinity of an interface there is considerable computational inefficiency. In an attempt to optimise computational efficiency Adalsteinsson and Sethian (1995) limited the extent of the level set function stored in the domain by creating a 'tube' of cells about the zero level set. The tube was created by calculating the distance from the curve for a given number of grid cells. The zero level set was then advected, as usual, with attention being paid to how close to the tube edge it had moved. Accuracy of the level set function was lost near

the edge of the tube so when the zero level set approached the edge of the tube or numerical instability occurred at the edge, the tube was reinitialised. Sethian (1996) improved the method by updating the list of tube cells about the interface as it moved. The method marked those points of the grid that were in the tube, outside the tube and close to the boundary of the tube. During each propagation step, the points inside and outside the tube were updated using the algorithm causing it to move with the zero level set. Peng *et al.* (1999) presented a based narrow band level set and reinitialisation procedure, by expanding on the method by Sussman *et al.* (1994). Numerical stability problems at the edge of the tube were overcome by reducing the speed of evolution of the level set function to zero near the tube edges. Furthermore, the method for updating the points within the tube, which points are to be added or removed, was replaced with an alternative method, further improving the efficiency of the algorithm.

Many authors have continued their research into more accurate methods of level set reinitialisation, thus addressing the mass loss problem. Russo and Smereka (2000) introduced a sub-cell fix to prevent sample points being used on the incorrect side of the level set function. This is a problem when upwind approximations are used close to the position of the zero level set. This method is straightforward to implement and minimises the erroneous shifting of the zero contour position. Hartmann *et al.* (2010) uses a constraint close to the zero contour along with high order approximation reinitialisation to correct any movement of the level set zero contour during reinitialisation using the original procedure. The method improves accuracy when using a WENO (Jiang and Peng, 2000) spatial approximation and allows for multiple reinitialisation iterations between time-steps without significant loss of mass. Unfortunately these methods are best suited to computation on structured grids.

Implementing level set re-distancing methods on unstructured grids has been a research interest in more recent times. Mut *et al.* (2006) takes a geometric approach to reinitialisation by sweeping the area immediate to the interface, the first neighbour nodes, calculating the distance away from it using a linear interpolation. The distance calculation is $O(h^2)$. An iterative volume preserving constraint is implemented to correct the interface to ensure it is not shifted based on the distance calculation. This volume preservation constraint is not dissimilar to that of Sussman *et al.* (1999). The iterative correction requires a sufficiently fine grid to obtain the best results. The method by Mut *et al.* (2006) was extended to curvilinear grids in Ausas *et al.* (2010). The method uses a decomposition of grid cells into triangles or tetrahedron (3D) to implement a geometric based reinitialisation method. This results in a computationally expensive method but is limited to certain formulations of unstructured grid made up of hexahedral cells. While the method seems to show improved accuracy at maintaining mass, work is required to extend the triangulation method for use on a more generalised unstructured grids.

The level set method offers a number of advantages over MAC and VOF methods. However, significant effort must be invested in ensuring the mass preservation of the simulated fluids. Ultimately the need to prevent movement of the zero level set contour has required the use of

volume preserving constraints and high order gradient approximations. Complex approximations lead to increased computational costs and difficult implementation for unstructured grids. In the following section some work is presented using hybrid schemes. Hybrid schemes aim to combine the benefits of MAC or VOF schemes with level set methods, to address some of the issues that are solved using high order spatial approximations.

2.2.4 Hybrid Methods

While 3rd order accurate TVD-RK time integration schemes and 5th order HJ-WENO approximations can provide an accurate and robust level set method, sometimes further accuracy is required. In an attempt to reduce mass loss to a minimum a number of authors have hybridised the level set method with either MAC or VOF methods. These combinations are aimed at incorporating the advantages of one method to account for the shortcomings of the other.

One of the successful hybrid VOF methods was published by Sussman and Puckett (2000) and called the Coupled Level Set/ Volume of Fluid (CLSVOF) method. The advantages that are offered are superior mass preservation, from the VOF method and improved geometric information at the interface, due to the implicit level set function. As Losasso *et al.* (2006) points out, mass loss due to truncation error of the volume fraction is several orders smaller than that due to level set reinitialisation.

The piecewise linear representation of the surface created by the VOF method can be used to initialise a level set function, at which point both the fluid fraction marker and level set function can be advected over a time-step. The superior geometric information provided by the level set function can be used to create a more accurate piecewise linear interface. Further to this it is possible to calculate surface curvature in a more robust, efficient manner than the piecewise linear interface representation.

Ship wakes were successfully simulated in Sussman and Dommermuth (2000) using a regular Cartesian mesh, however the simulation was only possible to achieve using a large parallel computer system. The authors suggest a move to adaptive grids may be necessary to resolve the problem efficiently. While others have had success with similar methods, such as Yang *et al.* (2006) on tetrahedral grids, the complexity of the routine make it prohibitively expensive to be considered the best solution.

Another hybrid level set method that takes advantage of a second interface tracking method is the Particle Level Set Method (PLS) introduced by Enright *et al.* (2002, 2003). In this case inert passive particles, somewhat like those of the MAC method, are seeded in a narrow band about the interface. The particles are passively advected with the solution and have a 'radius' equal to their distance from the free surface. When the advection of both level set function and particles is completed, the particle distances can be used to correct the level set function on a node by node basis. This correction need only occur in parts of the solution where the level

set function has been under resolved and significantly deviates from the particle solution. The particles can also be used to improve the reinitialisation step of the level set method. Finally, the particle radii are updated and the simulation can continue to the next time-step.

It was shown by Enright *et al.* (2003) that 1st order accurate advection and reinitialisation is a sufficient level of accuracy to manipulate the level set function during solution, yielding a simple cheap solution. It would be fair to say that the PLS method relies on the particles for the accuracy of the interface and the level set function simply for the connectivity of the interface. Due to the simplicity of the spatial and temporal approximations required to advect the components of the PLS method, it makes it a strong candidate for implementation on unstructured grids.

The particle level set method can suffer from particle bunching just like the MAC method requiring deletion and redistribution of particles through the domain. In areas of fluid break-up and coalescence this can require significant effort. An attempt to improve the PLS method, through simplification, was published by Mihalef *et al.* (2007) whereby marker particles were placed only along the zero level set contour. Particles and the signed distance function are then advected, with the particles helping correct the level set in areas of inaccuracy. While this method simplifies the PLS method by allowing for reseeding of particles at every time-step, the accuracy of tracking is reduced. Therefore this method is better suited to computer graphics applications rather than scientific simulations.

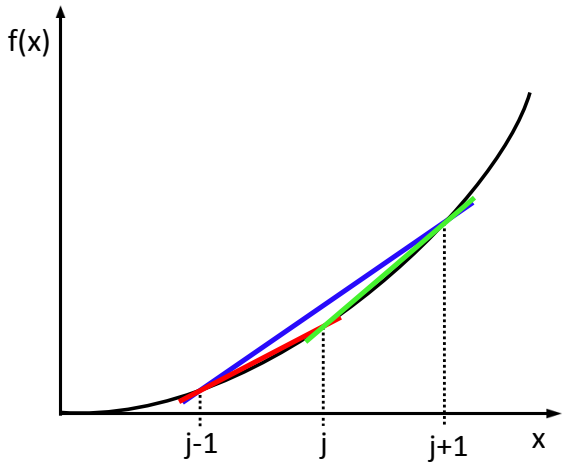


Figure 2.1: First order backward (red), forward (green) and second order central (blue) finite difference approximations of a high order function

2.3 Gradient Approximation and Finite Difference Modelling

When modelling differential equations, one of most universally accepted methods to obtain a solution is finite differencing. A finite difference approximation is an estimation of a derivative based on the difference of function's value at two points divided by the (finite) spacing separating them. The concept of finite differencing is based on the properties of Taylor expansions. Both (Hirsch, 2007, p147) and (Anderson, 1995, p128) cover this in some detail.

Given that the Taylor expansion is infinite, necessity dictates that the expansion be truncated at some position. Therefore all finite difference approximations suffer from a truncation error. This has a number of implications for numerical models of equations, the importance of these are discussed later in this section. The truncation of the Taylor expansion determines the order of accuracy of a finite difference approximation. An approximation with only the linear term of the Taylor expansion will not capture higher order terms that a function may contain. This leads to inaccuracy in the approximation if the function is quadratic of higher order. In most CFD applications, 1^{st} order accuracy is not sufficient to yield practically accurate results.

If a gradient approximation is required at a point, there are three simple ways of achieving this. The first is to take a difference between the point and its preceding neighbour and dividing by their spacing. The second, repeating the same procedure using the proceeding point and finally by taking a difference between the preceding and proceeding point, excluding the actual point in the calculation. These three approximations are referred to as backward, forward and central differences. The three different approximations are shown as red, green and blue lines in Figure 2.1.

Clearly, the different methods each give a different estimate of gradient at the point j. Through Taylor series analysis the forward and backward approximations can be shown to be 1^{st} order

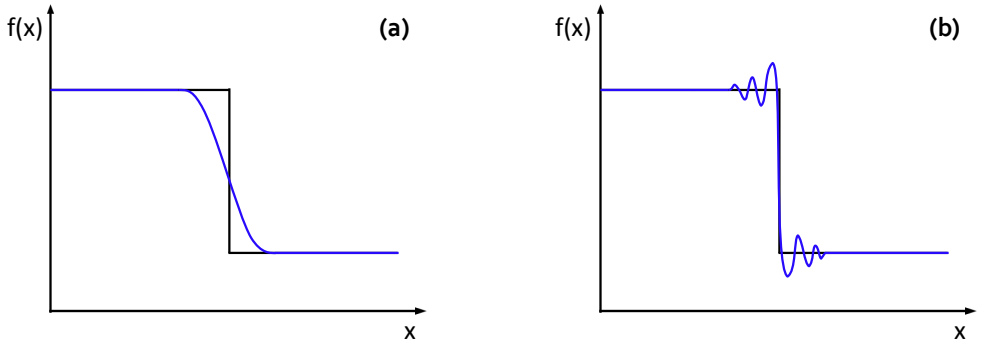


Figure 2.2: The numerical diffusion of a 1st order scheme (a) and the Gibbs phenomenon displayed by high order schemes (b). (Analytical solution black, numerical approximation blue)

accurate while the central approximation is 2nd order accurate. In the case of the curve in this example the higher order approximation leads to a more accurate estimate of the gradient. In addition, the accuracy of the approximations is increased as the spacing between points reduces to zero. In the vast majority of situations it is preferable to use a higher order accuracy scheme over a higher resolution of data points due to the increase in computational effort and memory. However, both low and high order numerical schemes suffer from numerical phenomena that have serious implications for numerical modellers.

Consider a one dimensional system where information is propagating from the left hand side to the right (advection). The propagating information is stored at discrete points equally spaced along the system axis. Given the direction of propagation, it makes physical sense to base finite difference approximations based on points that have had information already propagated to them. In this example the backward direction $j-1$ has already been updated with information and is commonly called the 'upwind' approximation due to the direction of propagation. In this case the forward or 'downwind' approximation has no influence on the propagation and therefore can only return an invalid answer. The central difference, taking information from both upwind and downwind can be used in areas of smooth data but has the potential to suffer from numerical instabilities where sharp changes occur.

In general approximating smooth data accurately is less challenging than data containing steep gradients or high curvature, areas become under-resolved due to poor point resolution and higher order schemes are required. If a discontinuity is introduced in the propagating information of the above paragraph a 1st order scheme and a higher order schemes may deal with it in different manners. Each scheme has advantages and drawbacks and is illustrated in Figure 2.2.

The left graph of Figure 2.2 represents the analytical solution and a 1st order numerical approximation of a discontinuity travelling through space after some arbitrary time. The numerical solution has 'smeared' the sharp discontinuity over some distance in space. This is simply due to the fact that if discontinuity falls between two points the maximum gradient that can be

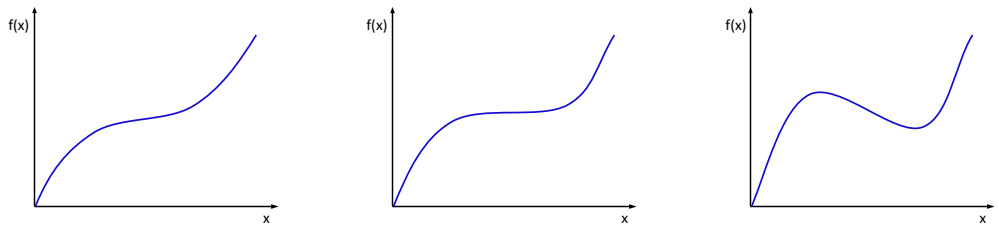


Figure 2.3: Strictly monotonic, weakly monotonic and non-monotonic functions

generated with a 1st order accurate scheme is the difference divided by a finite distance. As a discontinuity has infinite gradient, dividing by a finite distance will result in a less than infinite gradient. Over many time-steps, the extremely steep gradient is diminished until spread out as in the picture.

The right graph of Figure 2.2 represents the analytical solution and some high order numerical approximation. Here, the gradient of the discontinuity is more accurately preserved but oscillations about it are clear. These are known as Gibbs phenomenon and are produced as the result of using a 'stencil' of multiple points to produce the approximation. These oscillations can cause numerical instability and non-physical results, such as a negative density, to occur in a solution.

These properties play an important role in level set methods. Numerical diffusion contributes to a loss of the signed distance property, and accuracy of the method, especially in areas of high curvature where an interface may be under-resolved. Of equal importance for accuracy is maintaining the smoothness of the function in advection and during re-distancing. Numerical instability during coalescence and breakup of interfaces, where the level set function may contain steep gradients, is of particular importance.

The mathematical property that defines an ever increasing real function is monotonicity, Tveito and Winther (2005, p44). For a real function $f(x)$, it is strictly monotonic increasing if for $x > y$ then $f(x) > f(y)$. The same holds true for a strictly monotonic decreasing function where if $x > y$ then $f(x) < f(y)$. A relaxed version of this property is a weakly monotonic increasing function where for $x > y$ then $f(x) \geq f(y)$. Figure 2.3 gives examples of a strictly increasing monotonic function, weakly increasing monotonic function and non-monotonic function. It is highly desirable to maintain this monotonicity property in the level set function through simulation of numerical problems.

The major obstacle to finite difference modelling is maintaining a numerically stable solution while maintaining an accurate, high order approximation. Unfortunately due to the Gibbs phenomenon this is challenging in areas of high gradient or curvature. Godunov provided a landmark contribution to the field with his theorem, **"All linear monotone schemes for the advection equation are necessarily 1st order accurate"** (Godunov, 1959).

2.4 High Order Gradient schemes

A large volume of effort has been invested in producing high resolution gradient approximation methods that maintain the physical properties of the equations they represent. Section 2.3 highlights the fundamental principle and difficulty faced in numerical modelling, maintaining solution smoothness and accuracy without creating areas of instability or numerical diffusion in areas of high gradient. According to Godunov's theorem a numerical solution can only maintain monotonicity if approximated to 1^{st} order accuracy. Unfortunately the accuracy is insufficient, with non-physical numerical diffusion affecting the solution. There are many different approaches to avoiding confinement to only 1^{st} order accuracy but all follow the same principle - use a high order approximation in areas where the solution is smooth, identifying areas of steep gradient and switching to a lower order approximation to maintain stability.

In this section four categories of scheme are briefly outlined. The first two subsections cover two key scheme methodologies, those of flux corrected transport (FCT) and MUSCL (Monotone Upstream-centred Schemes for Conservation Laws) schemes. These schemes form the ideology and framework for the many schemes that have subsequently been developed. While a huge amount of work and research has been achieved using these schemes, most are suitable for application on structured grids.

FCT and MUSCL schemes were originally developed for compressible flows, a flow regime which depends on the stable, high resolution of discontinuities of pressure and density in the area of shock waves. These methods are equally important for simulating incompressible segregated fluid flow. A robust treatment is required for the jump condition in fluid density and viscosity, found at the free surface, especially if gross topological deformations are to be dealt with in a stable, accurate manner.

Essentially non-oscillatory schemes, have contributed to further improving the accuracy of CFD methods. They represent an important step forward in scheme accuracy with their ability to represent functions with high order accuracy even at discontinuities where gradient limiting is required. This is in contrast to TVD schemes which are limited to 2^{nd} order accuracy and whose solution tends to degenerate to 1^{st} order at discontinuities (Yue *et al.*, 2003).

A challenge of the 21^{st} Century has been to extend high order stable numerical methods to three dimensional unstructured grids. Schemes are regularly developed in 1D and extended to multiple dimensions, often through dimensional splitting, and where mathematical proof of their stability is impossible. In addition, due to the nature of unstructured grids it can be difficult to obtain data from the proper location for which to carry out limiting. It is for this reason "much of the recent and current research is oriented at the non-trivial extension of the concept of limiters to arbitrary unstructured grids" (Hirsch, 2007, p383). In the final part of the section some of the methods that have been implemented on unstructured grids are summarised.

2.4.1 Flux Corrected Transport Schemes

Flux Corrected Transport, commonly abbreviated to FCT, was introduced by Boris and Book (1973); Book *et al.* (1975); Boris and Book (1976). As one of the first finite difference methods that attempted to maintain high order accuracy where a function was smooth while limiting numerical diffusion and Gibbs phenomenon at steep gradients and discontinuities. The approach to achieving this was to explicitly add and subtract diffusive flux where a solution demanded it. It is stated by Kuzmin *et al.* (2005)[pVIII] "Almost all of the monotonicity preserving and non-oscillatory fluid transport algorithms of today trace their origins from flux corrected transport."

The first scheme to implement this was SHASTA (sharp and smooth transport algorithm) (Boris and Book, 1973). The scheme improved real world physical representation by enforcing non-negativity of mass and energy densities. This was achieved in a two stage calculation. First, fluid was transported using an Lagrangian style interpolation that was subject to a high degree of diffusion. Then, an anti-diffusive flux correction was calculated and applied to reduce the numerical errors due to diffusion. To preserve the monotonicity of the problem the anti-diffusion stage was limited to ensure no new extremes were created. The limiting of mass fluxes took place by comparing densities, ensuring that the correction stage did not move a density to exceed the value of its neighbours.

FCT is a technique and can therefore be applied to many numerical schemes. To this end Book *et al.* (1975) produced a generalised implementation of FCT for the Lax-Wendroff (Lax and Wendroff, 1960) and leapfrog schemes and further implementations are presented in Boris and Book (1976). Tuning of the diffusive and anti-diffusive fluxes, yielded large increases in accuracy over the non-FCT versions of these schemes. FCT was extended to a fully multidimensional form by Zalesak (1979) avoiding the need for a dimensional time-splitting procedure, prone to producing some poor numerical results.

One popular characteristic of FCT algorithms is their robust nature. They will generally tend to continue to solve a problem even if complications exist, rather than become unstable. This requires careful verification at the boundaries of the problem to ensure instability is not being corrected out (Boris *et al.*, 1993). Also artificial diffusion can remove solution fine detail as it smooths areas of the solution. Although subsequent algorithms have introduced their own constraints on the physics or mathematics of a numerical method, the concept of flux limiting still exists in algorithms such as MUSCL and TVD.

2.4.2 MUSCL Schemes

Bram Van Leer's approach to maintaining non-negative physical quantities (van Leer, 1974, 1977a,b, 1979) ultimately lead to the publication of the widely used MUSCL finite volume scheme (Monotone Upstream-centred Schemes for Conservation Laws). It was the first high order total variation delimited (TVD) scheme giving 2^{nd} order accuracy. Instead of using a piecewise constant approximation as per Godunov's scheme (Godunov, 1959), the flux is reconstructed using cell averaged states from the previous time-step. Therefore, a slope limited approximation can be achieved across each boundary of a cell.

Van Leer presented a modified monotonic version of the 2^{nd} order accurate upstream-centred Fromme scheme (van Leer, 1974). The modified Fromme scheme, Fromme monotonic scheme, is an average of two other schemes, the Lax-Wendroff scheme and an upwind scheme. It was shown that if both were modified to satisfy monotonicity, the average would also. The modified Fromme scheme had the advantage of being conservative, while preserving monotonicity and suffering from less diffusion than the 1^{st} order upwind Godunov scheme.

In order to better resolve real flow physics, Van Leer proposed an entirely upwind finite difference formulation of the Fromme scheme (van Leer, 1977a). The scheme was not particularly computationally attractive when compared to central difference schemes of similar accuracy. Indeed, the conclusion was that a higher order centrally differenced scheme could be employed to produce a more accurate answer for similar or less computational effort.

Adopting the Lagrangian approach of the Godunov scheme was introduced in van Leer (1977b). The initial value distribution of the Godunov scheme was replaced with a piecewise constant function. The function was easily transported and new cell averages calculated. The linear piecewise function was easily replaced with higher order approximations. However the use of higher order approximation required implementation of a flux limiter to achieve monotonicity. In this instance Van Leer achieved best success using the FCT method first introduced by Boris and Book (1973).

The series of work on conservative, monotonic compressible flow solvers finished with the introduction of MUSCL (van Leer, 1979). Of particular interest is the implementation of the slope limiter. Before convection, the piecewise representation was limited to ensure monotonicity. First the represented gradient was altered to ensure it was bounded by the neighbouring cell averages. If a mesh average reached an extrema, the gradient of the cells was reduced to zero. Finally, if a gradient did not agree with its neighbours trend the gradient was also reduced to zero.

2.4.3 Essentially Non-oscillatory Schemes

Essentially non-oscillatory (ENO) schemes are a class of approximation scheme that originate from the ideas developed with the MUSCL scheme of van Leer (1979) and the total variation diminishing schemes of Harten (1983); Sweby (1984) and Yee *et al.* (1985). Along with TVD and TVB (total variation bounded) schemes, ENO schemes preserve monotonicity without the need for artificial diffusion and provide high order accuracy (up to 5^{th} order) even locally to discontinuities, something that TVD schemes cannot claim.

ENO schemes use a high order interpolant to produce an upwind high order approximation without introducing spurious oscillations that are generated around discontinuities. To do so ENO schemes 'measure' the smoothness of the data surrounding a point, limiting the introduction of spurious oscillations by only basing their gradient approximation on interpolated data from smooth regions. While ENO schemes provide highly accurate numerical results there are very small interpolation overshoots that occur near extrema. This is due to the fact that a smooth polynomial function will slightly overshoot its values when sampled at discrete points near extrema. Because of this, the method is only *essentially* non-oscillatory.

Harten *et al.* (1987) introduced the ENO scheme, a piecewise linear polynomial reconstruction based on cell averages. The method employed an extensive grid stencil to help ensure that interpolation across discontinuous data was avoided. The scheme was improved in Harten (1989) with the introduction of sub-cell resolution with the aim of improving resolution and sharpness of linear discontinuities.

Harten's ENO schemes, based on conservative control volume discretisation, were complex and required transfer of values between cell averages and cell nodes. Additionally the method grew in complexity with each dimension added. Shu and Osher (1988) simplified the ENO scheme by composing the numerical flux functions directly from divided difference tables of point data, rather than cell averages. Further work by Shu and Osher (1989) improved the efficiency of the algorithm and introduced the methods of sub-cell resolution and artificial compression to improve the sharpness of contact discontinuities. Ultimately the method of Shu and Osher (1989) was adopted for the incompressible flow scheme with level set reinitialisation (Sussman *et al.*, 1994) that features significantly in this PhD thesis.

The article by Osher and Sethian (1988) was the first instance when an ENO scheme was extended to the solution of the Hamilton-Jacobi equations. ENO methods for the Hamilton-Jacobi equations are known as HJ ENO and allowed the accurate solution of level set methods for propagating fronts. The generalised HJ ENO method was published by Osher and Shu (1991) and included formulations extended from the existing methods of Shu and Osher (1989). These methods could yield up to 3^{rd} order accuracy results.

Liu *et al.* (1994) recognised that selecting a single gradient approximation from the three available using the ENO stencil was overly strict. Instead by weighting the combination of all three

candidate approximations the accuracy of the scheme could effectively be increased by an order in smooth areas. The weighting was constructed such that where approximations spanned a discontinuity, their contribution would be extremely small. This newer scheme was known as the weighted essentially non-oscillatory scheme or WENO. Following the work of Liu *et al.* (1994), Jiang and Shu (1996) was able to further improve the weighting method of the approximations, resulting in a WENO scheme that was more computationally efficient and an order of spatial accuracy higher.

Development of ENO schemes is continued to the present day, for example the work of Cho and Kim (2008) whose ENO scheme for the incompressible Navier-Stokes equations successfully simulates flows with high Reynolds numbers. The focus of modern research is toward efficient high order ENO schemes and schemes for arbitrarily unstructured grids. One of the principle restrictions of high order methods is their requirement for large stencils of grid points adding to the complexity of their implementation. Even AMR grids require solutions to be implemented either through hybridisation of gradient schemes (Kim and Liou, 2007) or through high order interpolation such as Min and Gibou (2007) where hanging nodes exist.

2.4.4 High Resolution Schemes on Unstructured Grids

While there has been a huge amount of successful research on high resolution gradient schemes, the volume of methods that have been implemented on unstructured grids is considerably less. Indeed the challenge for the CFD community in the early part of the 21st Century is to bring unstructured grid methods to a similar spectrum of applications as those for more conventional techniques. For unstructured grids the lack of grid points lying in the appropriate direction, for which limiting can be performed with, is a serious obstacle.

In his review of unstructured grid methods, Venkatakrishnan (1995a) comments that some early methods that were extended to unstructured grids tended to be extensions of one dimensional methods, implemented with dimensional splitting techniques, with little rigorous mathematical theory supporting them. They relied on a good quality, regular triangular mesh for their ability to capture shock interfaces with high resolution. Another difficultly faced on unstructured grids is that Riemann type solvers might misinterpret flow features that are not aligned with the cell faces due to the non-directional nature of unstructured grids.

One of the earliest methods to offer multi-dimensional limiting and reconstruction steps was a 2nd order Roe Riemann solver by Barth and Jesperson (1989). According to Darwish and Moukalled (2003) this has subsequently become one of the most popular TVD schemes for unstructured grids. Using a modified definition of the monotonicity criterion suggested by Spekreijse (1987), gradients were interpolated from cell centre to cell face and limited by neighbouring cell values to ensure no new extrema were introduced into the calculation. The scheme was a significant step forward as it works well on highly irregular grids. Unfortunately it is inappropriate for a large number of existing TVD schemes due to use of cell based gradients.

Subsequent work led to Venkatakrishnan (1995b) suggesting a limiter that improved solution convergence.

Jameson (1993) introduced a Symmetric Limited Positive (SLIP) scheme of the principle of Local Extremum Diminishing schemes for unstructured grids. LED was a multidimensional generalisation of the 1D TVD concept by Harten (1983) which relied on a non-linear limiter in all regions of the flow, not just close to discontinuities, hindering solution convergence in some cases (Mavriplis, 1997).

Jasak *et al.* (1999) extended a high resolution Normalised Variable Diagram (NVD) type solver introduced by Leonard (1988, 1991) to adaptive and unstructured grids. A reformulation of the NVD criterion for arbitrary geometry grids reduced the need for the use of far upwind data as it can be complex to establish which cell is an upwind cell with respect to the flow.

More recently, Darwish and Moukalled (2003) created a simple, generalised method, using the TVD criterion suggested by Barth and Jesperson (1989) so that it was applicable to a wider range of TVD schemes. This allows cell gradients to be generated using a weighted interpolation scheme, such as the least squares method, while still preserving the monotonicity conditions. Mandal and Subramanian (2008) have also used the least squares method to create what is termed as a solution dependent weighted least squares (SDWLS) method which incorporates a limiter to successfully produce a high resolution based finite volume method for both structured and unstructured grids.

Evidently, there has been a significant amount of work on the area of TVD schemes for unstructured grids, however there are some ENO schemes available for triangular unstructured grids. The difficultly in handling ENO schemes on unstructured grids is in part down to the large stencils that are required to implement them. Barth and Sethian (1998) introduced an ENO based scheme for level set methods on triangulated grids. However edge flipping was required to maintain the continuity of the solution and the mesh quality is a crucial in the accuracy of the solution.

More recently Zhang and Shu (2003) produced a WENO scheme for triangulated grids but the stencil for the grid area was quite large. This method uses a series of small cell stencils that make up a large stencil. The small stencils are combined with a weighting to produce a 3^{rd} or 4^{th} order accurate approximation. The method has been extended to fully 3D tetrahedral grids by Zhang and Shu (2009). While yielding more accuracy than the TVD schemes presented in this subsection, the stencil used is complex, requiring data from cells less than immediately adjacent. The potential difficulties this produces for boundary condition implementation is not insignificant.

2.5 Research Aim and Objectives

Free surface segregated flows are becoming an increasingly studied area of fluid dynamics (Qian *et al.*, 2003; Ingram *et al.*, 2009; Saruwatari *et al.*, 2009). While a number of different approaches to this class of problem have been developed, each has its own strengths and weaknesses. This is particularly true in problems where a smooth, continuous representation of the interface is required. In the case of simulating ocean waves the free surface boundary conditions can have a pronounced effect on the flow physics of the problem, especially when considering surface tension effects. This is apparent in the work of Saruwatari *et al.* (2009) whose study of breaking waves found the formation of finger jets to be strongly dominated by the surface tension. Furthermore, with the advances in Cartesian cut cell techniques (Ingram *et al.*, 2003), there are new opportunities to develop efficient and robust interface tracking methods for use on Cartesian cut cell grids.

For this PhD project, it was decided to develop a suitable interface tracking method that was capable of representing smooth, continuous surfaces on Cartesian cut cell grids. The interface tracking method chosen to implement is the level set method (Osher and Sethian, 1988). Its capability to represent fluid surfaces in a realistic manner, handle fluid break-up and coalescence and be implemented with a wide range of numerical schemes are all attractive qualities. The initial requirement is for a level set method with a reinitialisation procedure, in order to preserve the signed distance property of the function. To do so a reinitialisation procedure will be implemented and tested with an advection problem. Extension of the level set and reinitialisation methods is also possible, depending on future requirements.

In order to complete the development of the interface tracking method, an advection solver is required in order to transport test case interfaces about a domain. The advection solver that has been chosen for the evaluation of level set methods is the CIP method, Yabe and Aoki (1991); Yabe *et al.* (1991). It was chosen for its simple, robust high order explicit finite difference implementation. The CIP method has previously been used in areas of similar research implementing both a VOF scheme and level set method in the work of Watanabe and Saeki (2002); Watanabe *et al.* (2005, 2008); Saruwatari *et al.* (2009).

The main obstacle to using traditional level set reinitialisation methods with a Cartesian cut cell grid is the hybrid nature of the grid. While the majority of the grid is made up of a structured mesh, unstructured grid cells exist in the ‘cut’ cells. Therefore a gradient approximation method, suitable for use on both structured and unstructured grids must be sought.

Level set reinitialisation has traditionally used high order, upwind monotonic schemes to generate gradient approximations (Shu and Osher, 1989). Given the complexity of the popular ENO methods and the history of their implementation on unstructured grids (Barth and Sethian, 1998), it is difficult to guarantee success will be found attempting this approach. Furthermore, the ENO schemes that have been suggested for unstructured grids, are developed for grids with

triangular and tetrahedral elements, which are not the main constituent of Cartesian cut cell grids. Therefore a TVD approach to developing a scheme will taken.

One of the largest criticisms levelled at TVD schemes by others is the tendency for them to pollute or smear discontinuities. In the implementation presented here, this problem is of little concern. It is not expected that the normally smooth level set function will become discontinuous. The simplicity of a slope limited scheme's stencil and straightforward calculation currently outweigh any disadvantages when utilising them for re-distancing a level set function.

A summary of the objectives that must be satisfied to complete the aim of the project are as follows:

- Implement, test and evaluate a CIP advection scheme suitable for the transport of a level set function about a test domain.
- Implement, test and evaluate level set re-distancing methods and assess their impact on the accuracy of the level set interface tracking method.
- Design, describe, implement, evaluate and optimise a new method of gradient approximation, suitable for level set reinitialisation.
- Suggest a generalised implementation for use on a Cartesian cut-cell grid or other unstructured grid type.
- Compare the accuracy of the level set interface tracking schemes and determine the best scheme.

Chapter 3

The Cubic Interpolated Pseudo-particle Method

This Chapter introduces an advection solver known as the Cubic Interpolated Pseudo-particle method or the Cubic Interpolated Propagation method, the CIP method for short. The 1D and 2D methods presented in this chapter were originally presented in Yabe and Aoki (1991) and Yabe *et al.* (1991). The details and testing of the 1D CIP solver are covered in their entirety before covering a 2D solver. While a 3D version of CIP is described in the literature it is not required within the scope of this PhD project. Instead the reduced complexity of solving advection in a 2D framework is chosen for the development of the level set methods later in this Thesis. Strictly speaking the CIP method is a generalised hyperbolic equation solving method and will be effective for any equation that takes the form of

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0.$$

This is the form of a simple advection equation which is the major constituent of the LHS of the Navier-Stokes equations (Anderson, 1995, p75-77) and therefore an essential pre-requisite for solving fluid flow problems.

In this Thesis the CIP method will be used as an advection solver to transport a level set function about a computational domain in order to compare performance of level set interface tracking with a variety of level set reinitialisation schemes. However, first the CIP method must be implemented and correctly verified before results are compared with subsequent methods.

There are a multitude of methods for solving hyperbolic type equations. The key characteristics of a good hyperbolic solver are that it avoids numerical instability and has a low degree of numerical diffusion while not suffering from severe time-step restrictions. The CIP method is a universal solver for hyperbolic equations in one, two or three dimensions. The method is a sound choice for use as an advection solver, allowing for the development of level set methods. It provides a robust, low numerical diffusion, explicit finite difference method for solving the advection equation. In addition it allows equations containing both advection and non-advection parts to be solved in a two phases. This allows the developer to choose an ap-

propriate method for solving the non-advection phase independently of using the CIP method. This is, of course, an important consideration if the work in this Thesis is to contribute to some form of incompressible Navier-Stokes multiphase solver in the future. Work using the CIP method as part of a multiphase flow solver has been published by Yabe *et al.* (1998), Watanabe and Saeki (2002), Watanabe *et al.* (2005) and Watanabe *et al.* (2008) with the later three publications specifically using a level set method for interface tracking.

3.1 An Overview of the CIP Scheme

In order to maintain maximum solution fidelity with a high computational efficiency, schemes using Eulerian grids require some form of mesh resolution refinement in areas of high gradient. This has been done in multiple dimensions before by Berger and Colella (1989) but remains difficult nonetheless. Another method of maintaining solution fidelity on an Eulerian grid is to describe the profile at sub-grid resolution. The PIC method (Evans and Harlow, 1957) is one such method. Takewaki and Yabe (1986) identify PIC methods as somewhat inefficient methods in the manner they store information in both Eulerian and Lagrangian frames of reference. Nishiguchi and Yabe (1982, 1983) presented their SOAP method to address what were felt as some of the shortcomings of PIC. Unfortunately the high memory and computation requirements remained.

The Cubic Interpolated Pseudo-particle method is so called because the SOAP method was extended by Takewaki *et al.* (1984), to a single particle version using a cubic interpolated profile as a replacement for the costly Lagrangian particles that previously gave sub-cell resolution. By carefully taking measures, discussed in the section below, to ensure an accurate estimate of spatial derivatives to the advected quantity, the characteristic cubic equation can be solved without the need for costly matrix procedures. This ensures the CIP method is efficient and accurate in execution.

Section 3.2 covers the numerical algorithms of the CIP 0 and CIP 1 methods while Section 3.3 covers the boundary treatment and conditions used for all of the 1D test cases. Before verification and code testing, Section 3.4 provides an overview of code verification using the Grid Convergence Index method by Roache (1998). The CIP 0 implementation is verified and a comparison of performance for the CIP method with improved performance in transporting discontinuous functions (CIP 1) is compared with the basic CIP method (CIP 0).

The 2D CIP method is presented in the same manner as the 1D method, with Sections 3.7 and 3.8 describing the numerical algorithm and boundary conditions respectively. The verification and testing section is combined with a test case that is used throughout the remainder of the Thesis that will allow simple CIP 2D advection to be compared with various level set re-distancing methods.

3.2 The CIP Scheme Outlined in 1–Dimension

The methods presented in both this section (3.2.1) and Section 3.2.2 were first described in Yabe and Aoki (1991). This method is 3rd order accurate spatially and is implemented explicitly in time using a 1st order Euler time-step. In the case of explicit time formulations a Courant-Friedichs-Lewy (CFL) condition (Courant *et al.*, 1928) is a requirement for stable calculation.

In general terms, in a single dimension for an explicit scheme to remain stable, a 'particle' of fluid at a grid point cannot travel past the position of the next grid point in any single time-step.

$$0 \leq \sigma = \frac{u\Delta t}{\Delta x} \leq 1 \tag{3.1}$$

Equation 3.1 represents this in a mathematical form, where σ is the CFL number. The CFL number can be an important constraint to explicit methods, as methods requiring an extremely low CFL are forced to use a very large number of time-step calculations. Worse still, the criterion must be applied to the smallest cell with the highest velocity in the computational domain.

3.2.1 The CIP 0 Scheme

The function values at points on a grid can be determined at any instance of time by solving a one-dimensional hyperbolic equation, first consider it in the well recognised form below.

$$L_1 f \equiv \frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0. \tag{3.2}$$

Even if u is a function of time and space one method of expressing a local solution is

$$f(x_i,t) \sim f(x_i - u\Delta t,\ t - \Delta t), \tag{3.3}$$

where x_i is the grid point. Since the discretised value of f is only available at grid points $x_1,\ldots,x_i\ldots,f(x_i - u\Delta t)$ will be approximated by using these values. The cubic polynomial interpolation can be expressed as

$$F_i(x) = [(a_i\xi + b_i)\xi + f_i']\xi + f_i, \tag{3.4}$$

where $\xi = -u\Delta t$. If only the values f at all grid points are known, there are three parameters a_i, b_i, $f_i' = \partial f_i/\partial x$ to be determined. Conventional spline interpolation requires the continuity of f, $\partial f/\partial x$ and $\partial^2 f/\partial x^2$ at each grid point and this necessitates that complex matrix equations must be solved in order to determine a, b and f'. Such an approach is not suitable for efficient computation on vector computers and multidimensional cases further complicate the problem. Takewaki *et al.* (1984); Takewaki and Yabe (1986) show that the first spatial derivative must be determined consistently with the master equation.

Taking the spatial derivative of f from Equation. 3.2 gives a solution of

$$L_1(\partial f_x) = -\frac{\partial u}{\partial x}\frac{\partial f}{\partial x}. \tag{3.5}$$

If f, $\partial f/\partial x$ are given by Equations 3.2 and 3.5 and are assumed to be continuous at the grid point $i+1$, only two conditions are required to determine the coefficients a and b. As a result the coefficients are

$$\begin{aligned} a_i &= \frac{(f'_i + f'_{i+1})}{\Delta x^2} + \frac{2(f_i - f_{i+1})}{\Delta x^3}, \\ b_i &= \frac{3(f_{i+1} - f_i)}{\Delta x^2} - \frac{(2f'_i + f'_{i+1})}{\Delta x}. \end{aligned} \tag{3.6}$$

Once a_i and b_i are given in terms of f^n and f'^n, the value of f^{n+1} and f'^{n+1} are obtained by shifting the cubic polynomial in the same manner as Equation 3.3. Equation 3.3 is also of use for calculating f'^{n+1}. In the case of linear advection, where $u =$ constant, Equation 3.5 is equivalent to Equation 3.2. Thus the following equations express f^n and f'^n,

$$\begin{aligned} f_i^{n+1} &= F_i(x_i - u\Delta t) = [(a_i\xi + b_i)\xi + f'_i]\xi + f_i, \\ f_i'^{n+1} &= \mathrm{d}F_i(x_i - u\Delta t)/\mathrm{d}x = (3a_i\xi + 2b_i)\xi + f'_i. \end{aligned} \tag{3.7}$$

This expression is derived for $u < 0$. For $u \geq 0$, a similar expression is obtained by simply replacing Δx by $-\Delta x$ and $i+1$ by $i-1$ in Equation 3.6.

3.2.2 The CIP 1 Scheme

As Yabe and Aoki (1991) explain in Section 3 of their paper, the basic CIP 0 scheme is sufficient for many purposes including non-linear problems. The CIP 1 scheme increases the accuracy around discontinuities in a function. The CIP 1 scheme includes a special treatment for f' at the point where the spatial derivative is discontinuous. At a discontinuity, the edge of a square wave for example, the spatial derivative has different values, f'_R and f'_L, for the right sided and left sided spatial derivatives. This improvement to the scheme requires Equation 3.6 and Equation 3.7 to be rewritten as

$$\begin{aligned} a_i &= \frac{(f'_{i,R} + f'_{i+1,L})}{\Delta x^2} + \frac{2(f_i - f_i + 1)}{\Delta x^3}, \\ b_i &= \frac{3(f_{i+1} - f_i)}{\Delta x^2} - \frac{(2f'_{i,R} + f'_{i+1,L})}{\Delta x}. \end{aligned} \tag{3.8}$$

$$f_i^{n+1} = F_i(x_i - u\Delta t) = [(a_i\xi + b_i)\xi + f'_{i,R}]\xi + f_i,$$

$$f_i'^{n+1} = \mathrm{d}F_i(x_i - u\Delta t)/\mathrm{d}x = (3a_i\xi + 2b_i)\xi + f'_{i,R}. \tag{3.9}$$

For $u \geq 0$, a similar expression by simply interchanging the subscripts of R and L of f' in Equations 3.8 and 3.9 in addition to replacing Δx by $-\Delta x$ and $i+1$ by $i-1$ in Equation 3.8.

In all places except at the discontinuity, $f'_R = f'_L = f'$ and f' are estimated with Equation 3.9 as in CIP 0. However, at the discontinuity, $f'_R \neq f'_L$ and f' for this case is given below. In order to apply f'_R and f'_L appropriately the discontinuity must be detected. A new variable NP_i is created and set to $\mathrm{NP}_i = 0$, except at a discontinuity. The discontinuity is found by inspecting $|f_i - f_{i-1}|/|f_{i+1} - f_i|$, and NP is defined as

$$\begin{aligned} \mathrm{NP}_i &= -1 \quad \text{if} \quad \frac{|f_i - f_{i-1}|}{|f_{i+1} - f_i|} \quad \text{and} \quad \frac{|f_{i-1} - f_{i-2}|}{|f_{i+1} - f_i|} < \varepsilon \\ \mathrm{NP}_i &= 1 \quad \text{if} \quad \frac{|f_i - f_{i-1}|}{|f_{i+2} - f_{i+1}|} \quad \text{and} \quad \frac{|f_i - f_{i-1}|}{|f_{i+1} - f_i|} \geq \frac{1}{\varepsilon} \end{aligned} \tag{3.10}$$

ε should be as small as possible (Yabe and Aoki (1991) includes $\varepsilon = 0.05$ as a reference). At the discontinuity. f'_R or f'_L will not be determined from the master equation, but will be given by

$$\begin{aligned} f'_{i,L} &= \frac{f_i - f_{i-1}}{\Delta x} \quad \text{if} \quad \mathrm{NP}_i = -1, \\ f'_{i,R} &= \frac{f_{i+1} - f_i}{\Delta x} \quad \text{if} \quad \mathrm{NP}_i = 1. \end{aligned} \tag{3.11}$$

Yabe and Aoki (1991) presents further improvements to constraints used in the CIP algorithm. However it is at this point the two CIP methods considered in this PhD project are tested. During the results analysis, Section 3.6, discussion on the relative merits of the two methods is presented. Of note is the requirement for the CIP 1 scheme to fine tune the variable ε depending on the nature of the function being advected. While there is a recommendation for ε to be made as small as possible, at some point it becomes insensitive to sharp changes in gradient and ceases to perform as required, essentially reducing the scheme to a CIP 0 routine. The use of a larger number can prove to be over prescriptive in its application of slope limiting causing unnecessary numerical diffusion of the advected function.

3.3 1 – Dimensional Boundary Conditions

In the following sections numerical tests are conducted to characterise and assess the performance of the advection solver. In order to complete the tests there is a requirement for the function to be advected from left to right, leaving the right hand side of the computational domain and reappearing on the left hand side. There is therefore a requirement for a special set of conditions to be applied at the boundaries of the domain. In this case, the requirement is for a cyclic boundary condition to be applied. Furthermore, special treatment of numerical schemes is required for the outermost discretised points of the domain. Equation 3.6 contain terms f_{i+1} or f_{i-1}, of course it is impossible to complete the algorithm at the boundaries where these points do not exist.

The solution to this problem is to create additional invisible 'ghost' boundary cells that are available for use during the execution of the computer code but are invisible to the user when the data is output. For the CIP advection solver, two ghost cells are used at either end of the domain, the arrangement of which is represented in Figure 3.1. This allows a function, in this case the level set function, to be stored at all cells and the gradient of that function to be calculated to one point in from this. This arrangement allows the CIP method to advect information from the boundary cells into the real domain during the next time-step.

After a time-step is completed a boundary condition must be applied to maintain the validity of the data held in the ghost cells. In this case a cyclic boundary condition is applied. The boundary condition involves copying the data from the final cells at either end of the domain into the appropriate boundary cells so that data appearing at the RHS of the domain appears to move back into the domain from the left and, if the direction of transport is reversed, data from the LHS of the domain appears entering at the RHS. For the left hand side of the domain

$$f_{\text{b.c.1}} = f_{n-1} \quad \text{and} \quad f_{\text{b.c.2}} = f_n$$

and for the right hand side of the domain

$$f_{\text{b.c.3}} = f_1 \quad \text{and} \quad f_{\text{b.c.4}} = f_2$$

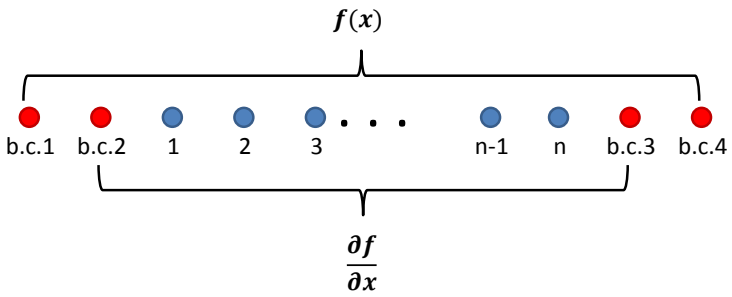


Figure 3.1: The arrangement of boundary cells in a 1D domain

3.4 Computer Code Verification

Once a computer code has been written it is necessary to test that is works in the expected manner. This section summarises the theory set out in Slater (2011) on the procedure for verification of CFD computer code. The verification methods here are used throughout this Thesis as numerical methods are presented. While a computer code might appear to function correctly, there is a need for a standardised procedure to report its performance. Of most immediate concern to the developer is that a computer code is verified to perform as the mathematical algorithm it represents. To the wider community, receiving a new numerical method, a standardised manner of reporting performance allows comparison between different numerical schemes.

The process of verification establishes that a computer code is capable of solving a system of coupled differential or integral equations, with initial conditions, correctly. The correctly functioning code should be able to reproduce the exact solution to a problem, given a grid of infinite (continuous) resolution is used and the convergence upon this exact solution should be observably of the same order as the methods used in the code. In this section a summary of a technique known as a grid convergence study (or grid refinement study) and its significance in CFD is presented.

A requirement for all CFD studies is to establish what level of grid resolution is required for a problem to be sufficiently resolved that an answer may be considered reliable. Due to the limitations of CPU speed, memory and data storage it is not practical or desirable to calculate a solution on a grid of virtual infinite resolution. When used as a design tool, a CFD study may consist of hundreds of simulations. It may be the case that highly accurate solutions are not required during the study or in all parts of the simulation domain. The appropriate grid resolution will depend on a number of factors, domain geometry, flow physics, the numerical schemes employed and boundary conditions in use. As with all approximate solutions, it is vital to estimate the error associated with it and whether the solution is reliable.

A grid convergence study begins with the generation of a solution on a relatively coarse grid. Subsequent results are generated on successively finer grids. It is expected that a working code will converge on the exact solution of a problem at close to the scheme's theoretical order of accuracy. Roache (1998) provided a widely adopted methodology for the uniform reporting of grid convergence studies. The method allows the user to generate an estimate of error, known as grid convergence index, on a grid based on the theory of Richardson extrapolation.

3.4.1 Considerations for Structured Numerical Grids

This short discussion on grid refinement strategies is limited to structured, orthogonal grids. However the grid refinement studies can be extended to unstructured grids in multiple dimensions if required.

One strategy of grid refinement is to choose the highest tolerable resolution, based on computational time, and subsequently remove alternating points to provide a coarser grid. If this is the case the number of grid points in each direction should be chosen using the formula

$$N = 2^n m + 1,$$

where N is the number of grid points in a dimension, n is the number of levels of coarsening or refinement possible, m is a base number equal to half the number of points minus one of the coarsest grid it is possible to build. Grids can be refined in individual dimensions or both together as is the case in all 2D problems throughout this Thesis.

The problems in this Thesis are generally quickly soluble with a modern CPU, none of which take significantly more than an hour of computation time. The grids throughout have been chosen to follow this guidance despite following a different strategy. The alternative strategy is to choose the coarsest grid and subsequently apply the formula to refine the resolution. Following this refinement method gives a refinement ratio of slightly less than 2, varying with the precise number of grid points used. However, a large number of test cases presented are symmetrical and the use of the formula provides a grid that always contains an odd number of points. This is of particular importance when dealing with a test function such as the one in Figure 3.2 in which the function's turning point is expected to begin and end advection in the centre of the grid. Without an odd number of points there is no point representing the turning point of the function, causing the maximum to be clipped. Therefore subsequent refinements would change the degree to which the function was clipped, changing the consistency of the solution with refinement. Assuming even grid spacing, following the above guidance results in each subsequent grid refinement adding an additional point between two existing points, thus maintaining the same grid point positions and placing a point at the function maximum.

3.4.2 Determining the Order of Grid Convergence

In order to provide accurate estimates of the error of a solution, the grids being used must be sufficiently refined that the solution error is within the asymptotic range of convergence. When this is the case C will produce a consistent value for successive grids.

$$C = \frac{E}{h^p}$$

A working CFD code's solution should converge upon the exact solution as the grid is refined. The numerical algorithms used should provide a theoretical order of accuracy that can be observed as the discretised answer converges on the exact solution. In practice a slightly lower observed order of accuracy may be recorded. This lower observed accuracy can be due to the influence of boundary conditions and the grid will reduce this slightly. Neglecting machine round-off error, the discretisation error can be written as

$$E = f(h) - f_{exact} = Ch^p + \text{H.O.T.},$$

where C is a constant, h is the grid spacing in each dimension, p is the order of convergence and H.O.T. are the higher order terms. If the higher order terms are neglected, then taking the logarithm of both sides of the equation and rearranging results in

$$\log(E) = p\log(h) + \log(C)$$

The order of convergence can therefore be obtained from the slope of the plot, $\log(E)$ vs. $\log(h)$.

An alternative to directly reading the gradient or calculating it by other means is to evaluate it using a constant grid refinement ratio,

$$p = \ln\left(\frac{f_3 - f_2}{f_2 - f_1}\right) / \ln(r),$$

where f_1 to f_3 is the discrete solution on successively coarser grids and $r = h_2/h_1$ is the refinement ratio.

3.4.3 Richardson Extrapolation

Having conducted simulations at successively fine grid resolutions, it is possible to obtain a higher order accuracy solution on an infinitely fine grid through a method called Richardson extrapolation using the lower order, discrete solution estimates. Whether or not the user chooses to do this depends entirely on his or her situation. However the theory used in the Richardson extrapolation is also the basis of the grid convergence index calculated by Roache (1998). The results from the Richardson extrapolation can be applied to the solution at every grid point

or to a quantities computed from the flow field such as drag. To do so, it is assumed that the solution is of uniform order globally as well as locally and that the quantity in question was also calculated using a method of the same order or better.

A simulation will yield a quantity f that can be expressed as a series expansion in general form

$$f = f_{h=0} + g_1 h + g_2 h^2 + g_3 h^3 + \ldots, \tag{3.12}$$

where g_1, g_2 and g_3 are functions independent of the grid spacing. If a p-order scheme is used to produce results for f on two different resolution grids and the higher orders term in the expansion in Equation 3.12 are neglected, the two resulting equations can be used to solve for $f_{h=0}$, the solution at zero grid spacing.

The generalised form for the Richardson extrapolation is for results of p-th order is

$$f_{h=0} \approxeq f_1 + \frac{f_1 - f_2}{r^p - 1},$$

where $r = h_2/h_1$ is the grid refinement ratio. It can be assumed that the Richardson extrapolation will yield an answer that is $p+1$ order accurate.

It is possible to take an estimate of error by taking the difference between $f_{h=0}$ and f_1, however as $f_{h=0}$ is predicted using a Richardson extrapolation the user must take into consideration the potential problems that arise. It is therefore more reliable to estimate error using the two code generated solutions, f_1 and f_2. The fractional error for generalised form of the Richardson extrapolation can be expressed as

$$A_1 = E_1 + O(h^{p+1}, E_1^2),$$

where $A_1 = (f_1 - f_{h=0})/f_{h=0}$. E_1 is the estimated fractional error of f_1 defined as

$$E_1 = \frac{\varepsilon}{r^p - 1}, \tag{3.13}$$

where ε is the relative error and is defined as $\varepsilon = (f_2 - f_1)/f_1$.

The relative error estimate for f_1 should not be used as the error estimate because it does not account for the refinement ratio and order of convergence and this can cause a poor estimate of the discretisation error. As the estimated fractional error, E_1, is an ordered error it will provide a better estimate of the grid error provided f_1 and f_2 are obtained with good accuracy. Good accuracy implies that the solutions are found to be within the area of asymptotic convergence with $E_1 < 1$.

If the Richardson extrapolation is expressed as,

$$f_{h=0} \approxeq f_2 + \frac{(f_1 - f_2) r^p}{r^p - 1},$$

it follows that an estimate of fractional error can be produced for the second grid in a study. This is written as

$$E_2 = \frac{\varepsilon r^p}{r^p - 1}.$$

Some of the disadvantages and caveats of using the Richardson extrapolation must also be taken into account as described by Roy (2005). As with any extrapolation, there is a tendency to amplify other sources of error such as round-off errors and, where applicable, iterative convergence error. In addition, an extrapolation does not have the same conservative properties that a finite volume scheme my have. Therefore in estimating a more accurate answer, it is possible to break the conservation property.

The underlying theory of Richardson extrapolation requires smooth solutions. It is possible to conduct a grid convergence study on a solution containing discontinuous data but the study will not be valid in that region. Furthermore if using a globally derived quantity the discontinuous data may reduce the effectiveness of the error estimates. In this Thesis the few test problems that contain discontinuous data are not subjected to error estimation in this manner. Finally, as the method requires multiple solutions in the asymptotic grid convergence range, solutions can be expensive to compute.

3.4.4 Grid Convergence Index

The method for error estimation used in this Thesis is the popular method by Roache (1998) known as the Grid Convergence Index (GCI). The GCI is a consistent method of reporting CFD results and provides an error band rather than an error estimate. The GCI can be calculated using two levels of grid, while three is preferred. The object is to provide a measure of uncertainty of the grid convergence. The GCI indicates how far a solutions is from the asymptotic value.

A factor of safety is added to Equation 3.13, yielding the GCI equation expressed in terms of percentage. This is recommended to be $F_s = 3.0$ for a GCI based on two grids and $F_s = 1.25$ for a GCI based on three. The GCI on the finest grid in the study is

$$GCI_{fine} = \frac{F_s|\varepsilon|}{r^p - 1} \times 100\%$$

and for the second grid is

$$GCI_{coarse} = \frac{F_s|\varepsilon|r^p}{r^p - 1} \times 100\%.$$

It is important only to estimate the error when grids have been established to be within the asymptotic range of convergence first. This can be achieved by checking values over three successively fine grids. An alternative measure of the asymptotic range coefficient is

$$C_{GCI} = r^p \frac{GCI_{12}}{GCI_{23}}.$$

A value close to one for C_{GCI} indicates that results are in the asymptotic convergence zone.

If one desires to refine each dimension of a problem individually, including time, perhaps with different refinement ratios, an independent coordinate refinement can be calculated by combining the GCI of each dimension individually to give an overall grid convergence index,

$$GCI = GCI_t + GCI_x + GCI_y + GCI_z.$$

A short FORTRAN program that performs the calculations outlined here, generating results for GCI, order of convergence and asymptotic range is available through the website Slater (2011). The program has been used widely throughout this Thesis generating the results tables that can be seen in sections which include grid convergence study results.

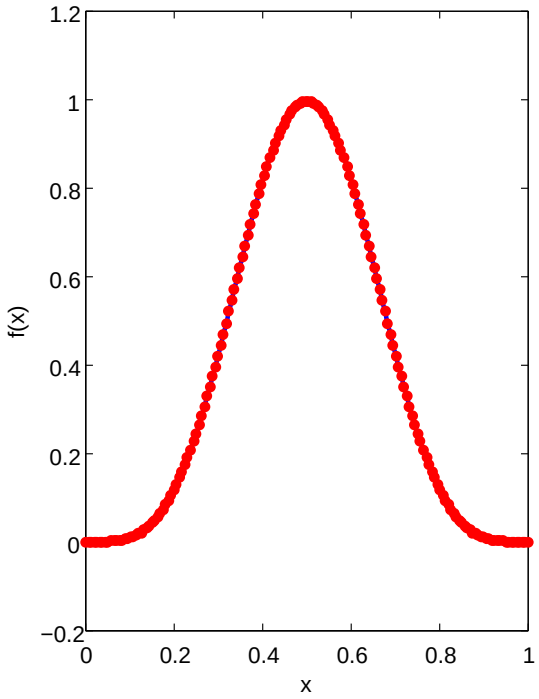


Figure 3.2: $f(x) = \sin^4(\pi x)$ advected for 100 cycles, 161 grid

3.5 CIP 1D Grid Convergence Study

A grid convergence study for the CIP 0 1D scheme is performed to establish an observed convergence rate of the computer code and verify convergence is to an exact solution. Unexpected or incorrect results indicate that there is an error in either the computer code or the implementation of the numerical algorithm. In order to provide data for the study, the function $f(x) = sin^4(\pi x)$ (Figure 3.2) is advected, left to right, for 100 cycles of the domain. To provide a global function for the grid convergence index, the standard deviation of the function is compared to the standard deviation of the analytical function. The high curvature found at the turning point of the sine curve creates a challenging area for the advection solver to resolve without adding a diffusion error. The smooth, continuous nature of the function, containing no discontinuous elements is ideal for the grid convergence study. These properties are also the key properties of the level set function.

Utsumi *et al.* (1997) demonstrated the CIP method is formally of 3rd order accuracy in space. For this study, the lowest resolution tested was 11 grid points. The grid was refined after each simulation was completed and analysed until solutions were found to be in the asymptotic range of convergence. The results are summarised in Table 3.1. Figure 3.3 plots $\log E$ vs. $\log h$, demonstrating the order of convergence visually.

Grid Resolution	Standard Deviation	GCI (%)	Order of Convergence	Asymptotic Range Ratio
Analytical	0.364434	—	—	—
161	0.364301	0.05	2.82	0.997525
81	0.363400	0.36	—	—
41	0.357026	—	—	—

Table 3.1: $f(x) = \sin^4(x)$ grid convergence results

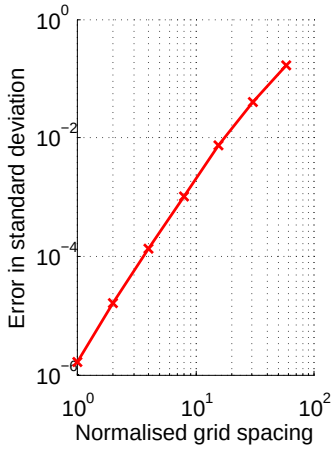


Figure 3.3: Normalised spacing vs. Standard deviation error for CIP 1D spatial study

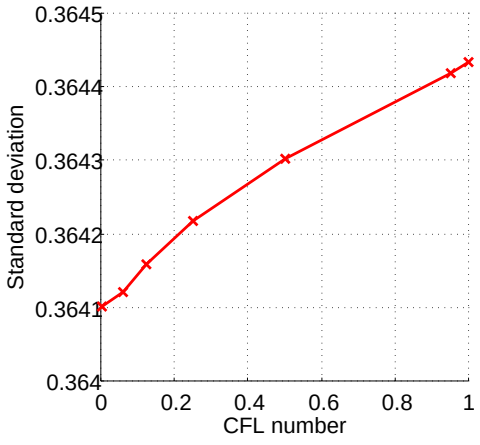


Figure 3.4: CFL number vs. Standard deviation solution for CIP 1D temporal study

The Richardson extrapolated result for the study was found to be 0.364450 demonstrating that the study converged on (or very close) to the exact solution. The observed order of convergence was calculated as 2.82 which is close to the theoretical order expected. These observations confirm that the CIP advection solver is implemented and working correctly.

Examining the CIP scheme performance while varying time-step size, through CFL number, there is no convergence apparent (Figure 3.4). An improvement in accuracy of the advected test function's standard deviation is observable up to a CFL number of 0.95. CFL numbers of 1.0 or higher are unstable, due to the function quantities being advected over the space of more than one grid cell per time-step. Clearly the reduced number of time-step calculations, due to the increased CFL number improves the accuracy of the 1D CIP method with smooth functions.

Yabe *et al.* (2004) writes that for a uniform velocity, as is the case for this test problem, the CIP scheme works out to be exactly conservative. It is therefore expected that performance of the CIP scheme will change significantly when implementing a non-uniform velocity field for 2D testing.

3.6 CIP Method 1D Numerical Tests

In this section, the original test cases conducted in Yabe and Aoki (1991) are repeated for further verification purposes and to allow a qualitative comparison between the CIP 0 and CIP 1 schemes. It is expected that the CIP 1 scheme will improve the tracking of the test functions at points were the data is discontinuous. There is also the opportunity to examine the effect of varying the CFL number on these simulations.

The first test presented is a piecewise function made up of a 'top hat' (Heaviside function) and a pyramid in 1-dimension. There are three principal areas of interest in the test function. The areas allow observation of overshoot or undershoot in the form of Gibbs phenomenon at the two discontinuous sides of the top hat function; numerical diffusion across the discontinuities of the top hat and clipping of the sharp turning point of the pyramid. All functions are advected from left to right on a 200 point cyclic grid for 5 cycles.

Figures 3.5 and 3.6 are reproductions of the CIP 0 and CIP 1 advection tests that appear in Yabe and Aoki (1991). The advected numerical solution can been seen as the red points over the analytical solution in solid blue. At the discontinuous sides of the top hat function a relaxation in gradient has occurred due to some degree of numerical diffusion. The solution has difficulty following the abrupt change in gradient at the ends of the discontinuity. Gibbs phenomenon are clearly visible in both the CIP 0 and CIP 1 solutions. The solvers track the sides of the pyramid accurately but slight oscillation of the numerical solution can be seen at the transitions either side due to the sharp change in gradient. Clipping of the point is slight but present nevertheless.

A direct comparison between the methods in the two figures shows similar results. The oscillations at the discontinuities of the top hat are reduced in the upstream direction while using the CIP 1 routine. The downstream oscillations remain similar in amplitude. The loss of gradient across the discontinuity is approximately the same for each scheme. The CIP 1 scheme does nothing to reduce the diffusion as the underlying interpolation scheme remains unchanged. While both schemes represent the pyramid with good accuracy, maintaining the transition between the flat of the function and the side of the pyramid reasonably well, it is possible to detect where the CIP 1 gradient limiter has been applied. The result is a small reduction in the oscillation on the upwind side of the pyramid. The sharp turning point of the pyramid is clipped slightly by the same amount in both schemes.

Continuing from the results of the grid convergence study in Section 3.5, a higher CFL number for the simulations may reduce the numerical diffusion effect. Immediately it was found that a high CFL number was unstable while using the CIP 1 method as soon as the slope limiter is applied. Therefore the low CFL number of 0.2 that is recommended by the original author was the limit for the CIP 1 results. The same problem is not encountered when increasing the CFL number of the CIP 0 scheme, in 1D at least. The advantage of the higher CFL is while oscillations remain, the diffusion across the discontinuities is visibly reduced. Another primary

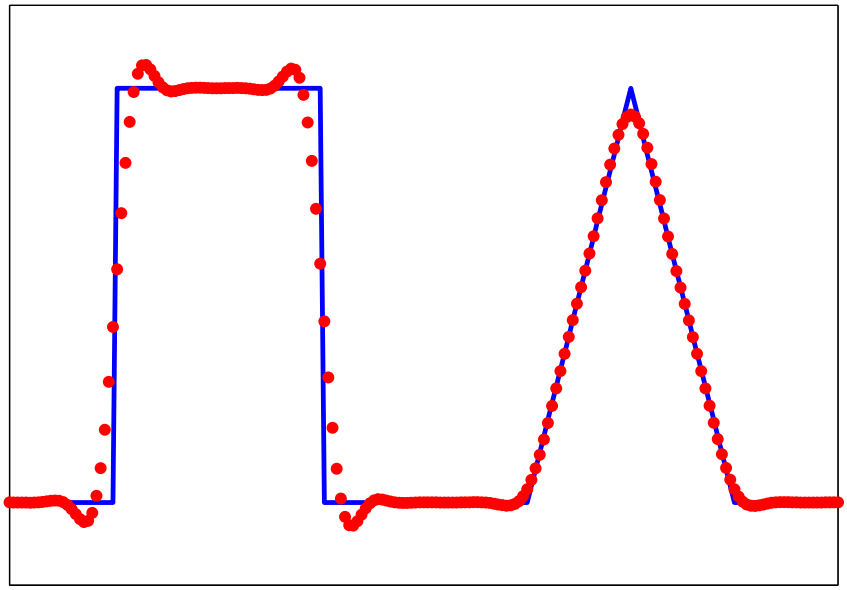


Figure 3.5: CIP 0 advection 1000 time steps, CFL=0.2

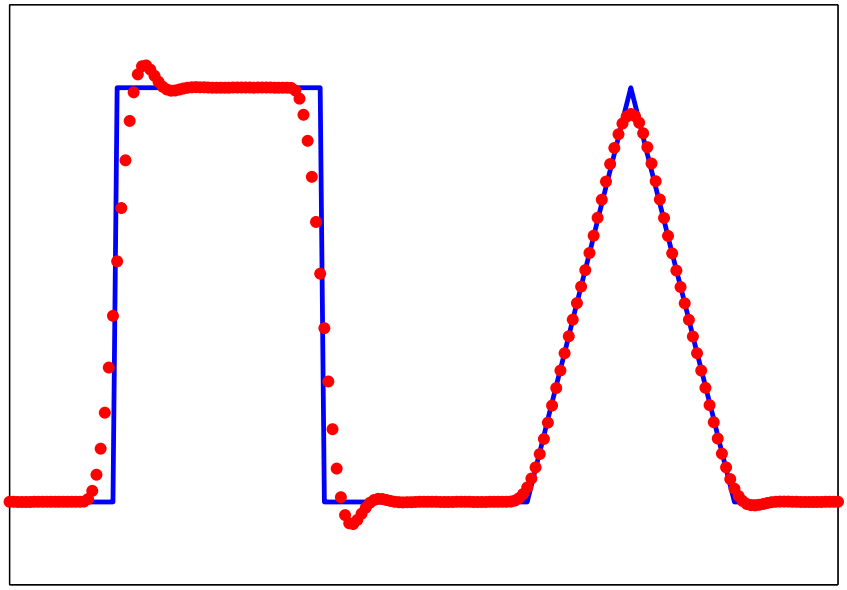


Figure 3.6: CIP 1 advection 1000 time steps, CFL=0.2

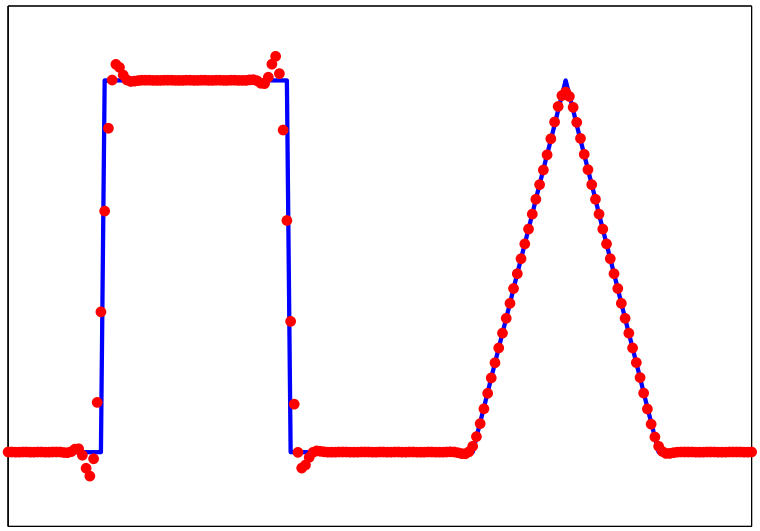


Figure 3.7: CIP 0 advection, CFL=0.95

benefit is the computational expenditure is reduced as the increased advection distance, due to larger time-steps, reduces the total number of time-steps that must be computed. Figure 3.7 is a repeated result of the CIP 0 solver using a CFL number of 0.95. It is clear that the diffusion on the solution is greatly reduced while the overshoot of the numerical artefacting remains similar.

The second qualitative test is of a half sine-wave advected using CIP 0 for the same time as the previous examples. This test function was chosen as due to the sign wave being a continuous, smooth function with an jump in it. This could be similar in situation as one moves across a fluid boundary in a simulation of two immiscible fluids. Figures 3.8 and 3.9 are of the simulation at CFL numbers of 0.2 and 0.95 respectively. Both sets of results show the numerical answer matching the analytical answer accurately where the function is smooth. However, the higher CFL number result shows less diffusion over the discontinuity and is therefore more accurate.

In conclusion, while the CIP 1 method does offer some reduction in numerical artefacting due to Gibbs phenomenon it does not completely eliminate it. When taking this into consideration along with a more constricted CFL number of around 0.2, the CIP 1 scheme does not seem that attractive. In the continuation paper covering 2D CIP methods, Yabe *et al.* (1991) have not included a version of CIP 1. When examining the CIP 0 method it is clear that for a continuously smooth function, the likely scenario to be encountered for the vast majority of incompressible flow, level set solvers, it performs adequately. It also would appear that in 1D a reduced number of time-steps, through a higher CFL number, reduces numerical diffusion, provided stability can be maintained.

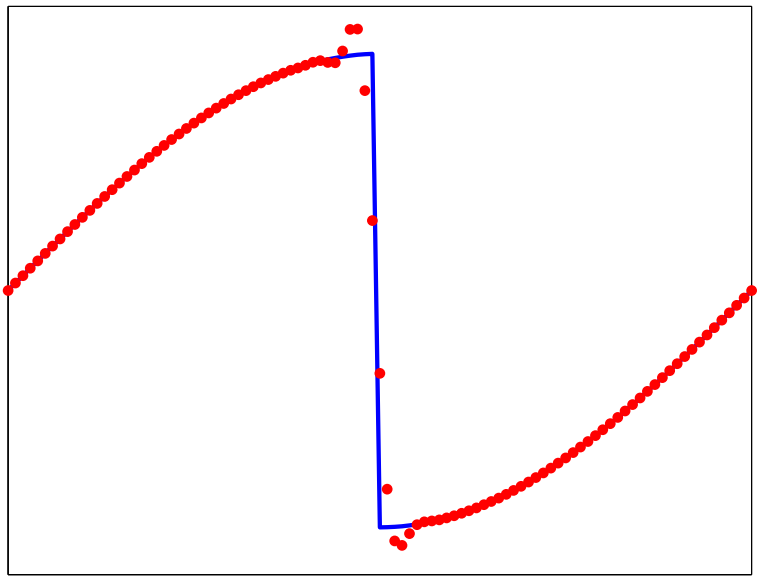


Figure 3.8: CIP 0 advection CFL=0.2

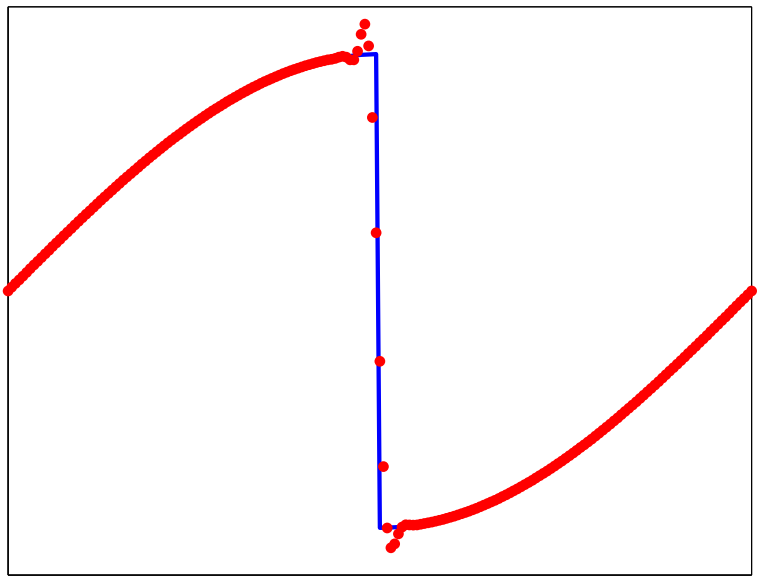


Figure 3.9: CIP 0 advection CFL=0.95

3.7 A Numerical Outline of the CIP Scheme in 2-Dimensions

Moving forwards, a 2D advection solver is required for testing level sets effectively. Immediately below, a method for 2D CIP is presented. The algorithm is the work of Yabe *et al.* (1991) and is a CIP 0 method. It does not contain a slope limiter procedure such as the one described in Section 3.2.2. This chapter concludes with shapes, represented by a zero level set contour, being moved around a 2D computational domain. Section 3.9 will demonstrate and highlight the need for additional methods to maintain the signed distance property of the level set function, resulting in more accurately tracked geometries.

When discussing the CIP method in a 2D reference frame, the spatial dimensions are x and y and components of velocity are u and v respectively. For two dimensions, the hyperbolic equation, similar to Equation 3.2, is extended,

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + v\frac{\partial f}{\partial y} = 0. \tag{3.14}$$

For an orthogonal grid with constant spacing Δx and Δy, using the same logic as the 1D method, when $u < 0$ and $v < 0$, $f(x,y)$, f is interpolated over a grid cell with a cubic polynomial,

$$\begin{aligned} F_{i,j}(x,y) = &[(\mathrm{A1}_{i,j}\xi + \mathrm{A2}_{i,j}\eta + \mathrm{A3}_{i,j})\xi + \mathrm{A4}_{i,j}\eta + \partial_x f_{i,j}]\xi \\ &+[(\mathrm{A5}_{i,j}\eta + \mathrm{A6}_{i,j}\xi + \mathrm{A7}_{i,j})\eta + \partial_y f_{i,j}]\eta + f_{i,j}. \end{aligned} \tag{3.15}$$

where $\xi = -u\Delta t$ and $\eta = -v\Delta t$. If the values for f at all grid points, $f_{i,j}(i = 1,\ldots,i_{max}, j = 1,\ldots,j_{max})$, are known, there are nine parameters A1, ..., A7, $\partial f/\partial x$ and $\partial f/\partial y$ that must be determined.

Again, the first spatial derivatives must be determined consistently with the master equation. This will allow an accurate solution to be obtained without solving a matrix equation to determine the cubic polynomial. The equations for the first spatial derivatives from Equation 3.14 are

$$\begin{aligned} \frac{\partial f}{\partial x} &= -\frac{\partial u}{\partial x}\frac{\partial f}{\partial x} - \frac{\partial v}{\partial x}\frac{\partial f}{\partial y}, \\ \frac{\partial f}{\partial y} &= -\frac{\partial u}{\partial y}\frac{\partial f}{\partial x} - \frac{\partial v}{\partial y}\frac{\partial f}{\partial y}. \end{aligned} \tag{3.16}$$

For simplicity, in the following equations the notation of the spatial derivatives of f are switched to $\partial_x f$ and $\partial_y f$. If f, $\partial_x f$ and $\partial_y f$ at all grid points are given by Equations 3.14 and 3.16, seven conditions are required to determine the coefficients A1, ..., A7. Although there may be a number of choices for these conditions, just as before, f and the spatial derivatives, $\partial_x f$ and $\partial_y f$, must be continuous at the grid points $(i+1, j)$ and $(i, j+1)$ and f must be continuous at the

grid point $(i+1, j+1)$. As a result the following equations are obtained:

$$\begin{aligned}
\mathrm{A1}_{i,j} &= [-2d_i + \partial_x(f_{i+1,j} + f_{i,j})\Delta x]/\Delta x^3,\\
\mathrm{A2}_{i,j} &= [\mathrm{A8}_{i,j} - \partial_x d_j \Delta x]/\Delta x^2 \Delta y,\\
\mathrm{A3}_{i,j} &= [3d_i - \partial_x(f_{i+1,j} + 2f_{i,j})\Delta x]/\Delta x^2,\\
\mathrm{A4}_{i,j} &= [-\mathrm{A8}_{i,j} + \partial_x d_j \Delta x + \partial_y d_i \Delta y]/\Delta x \Delta y,\\
\mathrm{A5}_{i,j} &= [-2d_j + \partial_y(f_{i,j+1} + f_{i,j})\Delta y]/\Delta y^3,\\
\mathrm{A6}_{i,j} &= [\mathrm{A8}_{i,j} - \partial_y d_i \Delta y]/\Delta x \Delta y^2,\\
\mathrm{A7}_{i,j} &= [3d_j - \partial_y(f_{i,j+1} + 2f_{i,j})\Delta y]/\Delta y^2,\\
\mathrm{A8}_{i,j} &= f_{i,j} - f_{i+1,j} - f_{i,j+1} + f_{i+1,j+1},
\end{aligned} \tag{3.17}$$

where $d_i = f_{i+1,j} - f_{i,j}$ and $d_j = f_{i,j+1} - f_{i,j}$. The profile after a time-step can be approximated by,

$$\begin{aligned}
f_{i,j}^{n+1} = [(\mathrm{A1}_{i,j}\xi + \mathrm{A2}_{i,j}\eta + \mathrm{A3}_{i,j})\xi + \mathrm{A4}_{i,j}\eta + \partial_x f_{i,j}]\xi\\
+[(\mathrm{A5}_{i,j}\eta + \mathrm{A6}_{i,j}\xi + \mathrm{A7}_{i,j})\eta\partial_y f_{i,j}]\eta + f_{i,j}^n,
\end{aligned} \tag{3.18}$$

$$\partial_x f_{i,j}^{n+1} = (3\ \mathrm{A1}_{i,j}\xi + 2\ \mathrm{A2}_{i,j}\eta + 2\ \mathrm{A3}_{i,j})\xi + (\mathrm{A4}_{i,j} + \mathrm{A6}_{i,j}\eta)\eta + \partial_x f_{i,j}, \tag{3.19}$$

$$\partial_y f_{i,j}^{n+1} = (3\ \mathrm{A5}_{i.j}\eta + 2\ \mathrm{A6}_{i.j}\xi + 2\ \mathrm{A7}_{i,j})\eta + (\mathrm{A4}_{i,j} + \mathrm{A2}_{i,j}\xi)\xi + \partial_y f_{i,j}. \tag{3.20}$$

Both the 1D and 2D CIP solvers can be used to solve a more generalised hyperbolic equation,

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} + v\frac{\partial f}{\partial y} = g.$$

The equation is split into two phases, the LHS being the advection phase and the RHS being the non-advection phase. A variety of methods can be used to solve the non-advection phase, depending on the specifics of the problem. A situation such as this arises when trying to solve the incompressible form of the Navier-Stokes equations for fluid flow. Authors such as Watanabe and Saeki (2002) have used the CIP method for precisely this purpose. A description of how to proceed in this case is included in both Yabe and Aoki (1991) and Yabe *et al.* (1991). For the purposes of the PhD project only the advection phase is required, and performance of the advection method is tested in 2D with a grid convergence study.

3.8 2–Dimensional Boundary Conditions

For the two dimensional test problems presented in this Thesis a rotating velocity field, initialised about the centre of the domain is used. While the velocity field is fixed and requires no updating at the boundaries, the level set function must be handled in an adequate manner such that the boundaries of the problem do not affect the solution within the domain.

The boundary conditions are applied at the end of each time-step calculation. Because the level set function is a smooth, continuous function a simple linear projection is used to update the boundary cells at the edges of the domain. The projection is based on the gradient of the last two real domain cells and projected to the boundary cells.

Following the notation for boundary cells in Section 3.1, with $f^x_{b.c.1}$ and $f^y_{b.c.1}$ representing boundary cells on the rows of the numerical grid and columns of the grid respectively.

For the boundary cells along the rows, down the sides of the domain

$$D_x(f_1) = \frac{f_2 - f_1}{\Delta x} \qquad D_x(f_{ni}) = \frac{f_{ni} - f_{ni-1}}{\Delta x}$$

$$f^x_{b.c.2} = f_1 - D_x(f_1) \times \Delta x \qquad f^x_{b.c.1} = f_1 - D_x(f_1) \times 2\Delta x$$

$$f^x_{b.c.3} = f_1 + D_x(f_{ni}) \times \Delta x \qquad f^x_{b.c.4} = f_1 + D_x(f_{ni}) \times 2\Delta x$$

and for the boundary cells across the top and bottom of the domain

$$D_y(f_1) = \frac{f_2 - f_1}{\Delta y} \qquad D_y(f_{nj}) = \frac{f_{nj} - f_{nj-1}}{\Delta y}$$

$$f^y_{b.c.2} = f_1 - D_y(f_1) \times \Delta y \qquad f^y_{b.c.1} = f_1 - D_y(f_1) \times 2\Delta y$$

$$f_{b.c.3} = f_1 + D_y(f_{nj}) \times \Delta y \qquad f^y_{b.c.4} = f_1 + D_y(f_{nj}) \times 2\Delta y$$

3.9 2-Dimensional Grid Convergence Study

Just as for the CIP method in 1D, the performance of CIP in 2D can be assessed using a grid convergence study. The test case uses a level set function, reviewed in Section 2.2.3, whose zero isocontour will represent a meaningful shape of known area. The numerical properties of the level set method will be discussed in detail in Chapter 4. Conducting a grid convergence study ensures the scheme not only guarantees consistent spatial and temporal convergence, but also provides a grid resolution which adequately resolves the test problem on which subsequent comparative tests using level set reinitialisation routines can be run.

For the PhD project, a simple test problem was chosen to allow for straightforward analysis of the numerical behaviour of the problem. In addition, it is desirable that the solution of the test case would take minutes not hours, allowing for efficient analysis of the problem in convergence studies. Therefore a problem that could be resolved on a grid of moderate size was chosen.

The test case uses a level set function initialised on a uniform square grid with a circular zero contour. The limits of the grid are $[-0.5, 0.5]$ and the circular contour is initialised at the 12 o'clock position on the grid with its centre at $[0.0, 0.25]$. The diameter of the contour is 0.2 and as such, the area contained by the contour can be calculated precisely as exactly $\pi \times 0.1^2$. By initialising a circular clockwise velocity field about the centre of the numerical grid, the level set function can be circulated indefinitely. By measuring the initialised contour's area and then at subsequent time steps, using the algorithm presented in Appendix B, it is possible to establish how numerical errors distort the contour and reduce its area.

Three revolutions of the numerical domain were simulated, allowing simulation on an extremely coarse 25×25 grid without the contour disappearing from the simulation due to numerical diffusion. The advected distance also kept simulation within acceptable time limits on the highest resolution grid. The highest CFL number that remained stable on the finest and highest resolution grid was found, through experimentation, to be 0.5.

The results of a simulation on the 25×25 coarse grid can be seen in Figures 3.12 and 3.13 demonstrating the area loss that can occur using level set interface tracking where the grid is under-resolved. There are three features of results in this diagram of note. Before advection, the grid resolution is so low that it clearly affects the quality of the shape of the initialised contour. While the contour function is circular the grid resolution is so low it affects the quality of the shape, with visibly flattened sides, before advection takes place. This occurs to a lesser degree on any grid initialised function and accounts for a discrepancy in area between the initialised contour and the analytical answer. After advection, area loss due to numerical error has reduced the contour area to under 6% of its starting value. Finally, the position of the contour's centre point is offset from its original starting position. Clearly a solution such as this is unacceptable to any user but these phenomena affect all level set simulations to a greater or lesser degree in

regions where the function is under-resolved.

Grid Resolution	Contour Area	GCI (%)	Order of Convergence	Asymptotic Range Ratio
Analytical	3.141593×10^{-2}	—	—	—
769×769	3.049665×10^{-2}	3.89	0.96	0.970277
385×385	2.959021×10^{-2}	7.83	—	—
193×193	2.781723×10^{-2}	—	—	—

Table 3.2: 2D cylinder grid convergence results

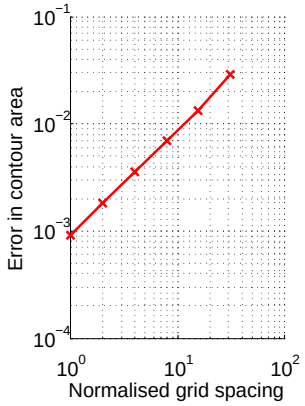


Figure 3.10: Contour area spatial convergence

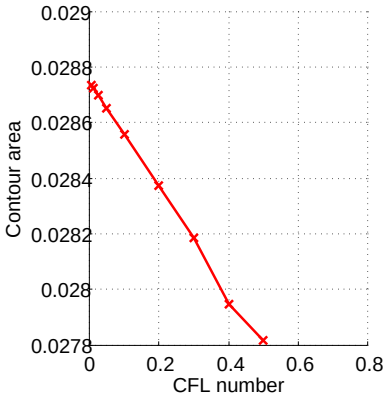
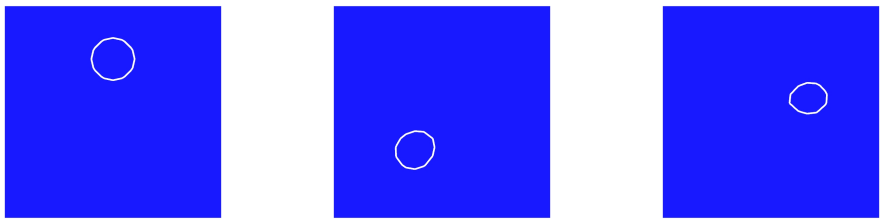


Figure 3.11: CFL number vs. Contour Area as part of the grid convergence study

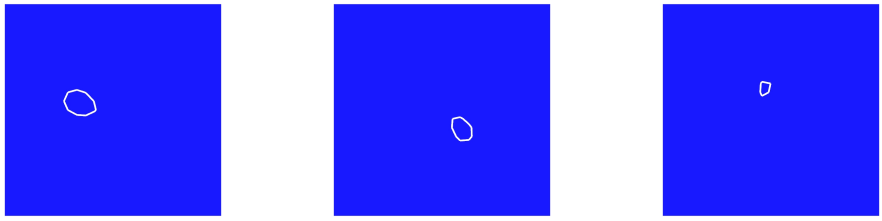
Table 3.2 contains the results of the spatial grid convergence study. The asymptotic range ratio, calculated over three grids as an indication of grid convergence, is required to be close to 1 to indicate convergence inside the asymptotic range. The ratio becomes close to 1 after a grid resolution of 193×193 grid points and remains constant at approximately 0.97 at resolutions over 385×385 grid points. Figures 3.14 and 3.15 show the improvements in accuracy from the under-resolved grid. Therefore an answer simulated on the second finest grid is sufficient to produce a solution with an uncertainty, based on the GCI, of $\pm 7.83\%$. The accuracy of the simulation can be further increased, as desired, by increasing the resolution of the grid but also this increases the computing time and memory requirements of the simulation. In the extensions to this test problem, the focus is on improving the accuracy of this test on grids of the same resolution.

A short spatial and temporal convergence study was also conducted, assessing the effect of the choice of CFL number in the accuracy of the simulation. Using 1^{st} order explicit time-stepping the results show that a reduction in CFL number improves the accuracy of the advection problem. The graphs in Figures 3.10 and 3.11 illustrate these results. This may seem surprising when a comparison is made with the 1D results, in which increased CFL number improved the accuracy of the results. It is likely that in 1D, as the CFL number approached a value of 1, the solution simplified towards an exact solution. Because only a single dimension is involved, a CFL number of 1 causes the value held at a point to be moved exactly to the next grid point, resulting in no loss of information. In this 2D test case, function values are advected horizontally, vertically and diagonally across the grid due to a rotational velocity field. Function values are not likely to be moved precisely to an adjacent grid point in this case. Less error is introduced with a reduced CFL number as the shorter interpolation steps lead to less numerical error being introduced at each time-step.

Sufficient information has been collected using the circular 2D cylinder test case to have the confidence that the CIP 2D advection routine is working correctly and forms a good basis with which to compare improvements in accuracy brought about when introducing a level set re-distancing scheme into the solver. Further detail of the level set method and level set reinitialisation is introduced in the following chapter.



(a) 0% of simulation **(b)** 20% of simulation **(c)** 40% of simulation



(d) 60% of simulation **(e)** 80% of simulation **(f)** 100% of simulation

Figure 3.12: CIP advection, 25×25 point grid, CFL=0.5, 3 clockwise revolutions of domain

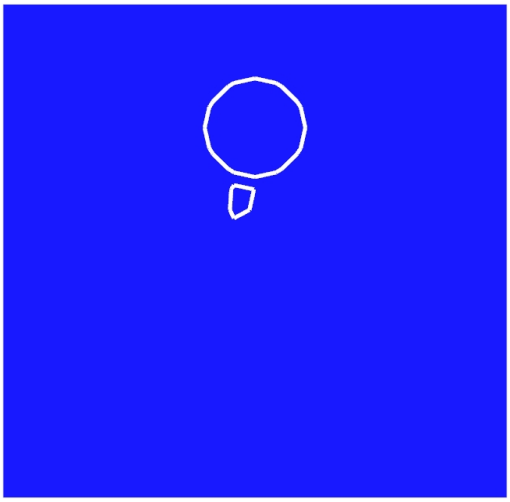
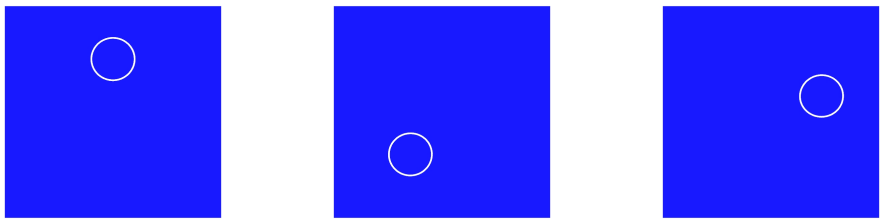
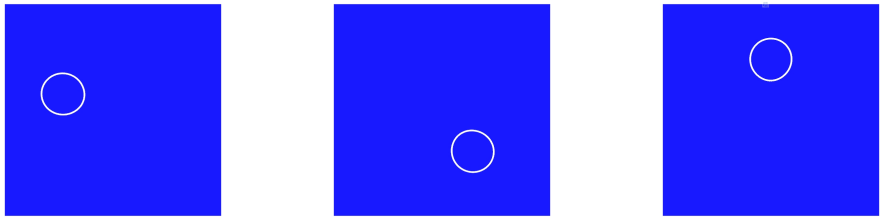


Figure 3.13: CIP advection, 25×25 point grid, CFL=0.5, start/ finish comparison



(a) 0% of simulation **(b)** 20% of simulation **(c)** 40% of simulation



(d) 60% of simulation **(e)** 80% of simulation **(f)** 100% of simulation

Figure 3.14: CIP advection, 385×385 point grid, CFL=0.5, 3 clockwise revolutions of domain

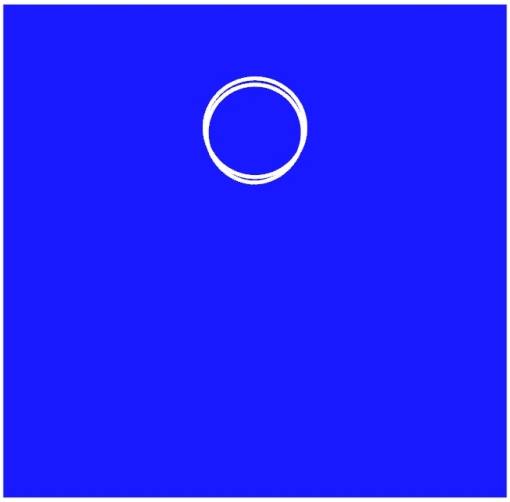


Figure 3.15: CIP advection, 385×385 point grid, CFL=0.5, start/ finish comparison

3.10 Concluding Remarks on the CIP Method

In this chapter both CIP 0 and CIP 1 methods were presented, tested and evaluated. In the case of the 1D tests, the CIP 1 method successfully suppressed some of the Gibbs oscillations in areas of sharp gradient change. However, oscillations were not totally suppressed using the CIP 1 method. It was also noted that an awkward feature of the CIP 1 method is the tuning of the parameter ε, of Equation 3.10, which controls the implementation of the slope limiter. At small values the parameter was not sensitive and failed to suppress any oscillations. When the value of ε was chosen to be too large then it unnecessarily smeared the advected solution.

For the case of level set advection that is being analysed in this Thesis, the CIP 1 method does not add a great benefit when considering the advection of a smooth signed distance function. Therefore the 2D case was extended using the CIP 0 method.

For the test case of a circular contour being advected around a square domain for three revolutions, a grid resolution of 385×385 was required to produce a fully converged solution. Even in the case of the fully converged result some area loss of the contour was evident. In under resolved solutions severe area loss and shape distortion occurs. The next chapter will introduce a re-distancing method called level set reinitialisation. Using reinitialisation the signed distance of the level set function can be preserved more accurately leading to improved results.

Chapter 4

The Level Set Reinitialisation Procedure

4.1 An Overview of Level Set and Reinitialisation Methods

Chapter 3 showed that as a level set function is advected, using Equation 3.2, numerical diffusion acts upon it, causing it to lose its signed distance property. In addition to this, in a scenario where the background velocity field is non-uniform, for example an incompressible flow simulation, the local velocities across the domain will 'jumble' the level set function and further accelerate the loss of the signed distance property. Before continuing by demonstrating a popular technique that can be used to improve level set interface tracking, some more mathematical detail of this function and its properties are outlined.

Representing an interface explicitly involves tracking a number of points that lie on the interface. Previously discussed in Section 2.2, this is not necessarily an accurate or computationally efficient method of interface tracking. Rapidly deforming interfaces that may suffer break-up and coalescence are especially difficult to follow. An alternative, choosing to represent an interface implicitly, relies on defining the interface using the iso-contour of some function.

To illustrate this, consider a one dimensional domain, shown in Figure 4.1. The single spatial dimension x has been divided into the subdomains $\Omega^- = (-2,2)$, where $\phi(x) < 0$ is Ω^-, and $\Omega^+ = (-\infty,-2) \cup (2,\infty)$, where $\phi(x) \geq 0$ is Ω^+. The function $\phi(x) = x^2 - 4$ can be used to define interfaces, in this 1D case as points $\partial\Omega$ between the subdomains. The zero iso-points can be used to represent two interfaces that define the boundaries between subdomains with $\partial\Omega = \{-2,2\}$. In general, any isocontour of the appropriate function can be used to represent the interface, however, convention has set the zero isocontour as the interface. When the level set function is described as a signed distance function it becomes clear that this is a logical choice.

Increasing the domain to a two dimensional case, an isocontour can represent a line interface between two subdomains, whose area is greater than zero. In Figure 4.2 is a closed curve isocontour representing an interface between subdomains. Inside the zero contour the function $\phi < 0$ and outside $\phi > 0$. In this simple example there is no great barrier to using an explicit

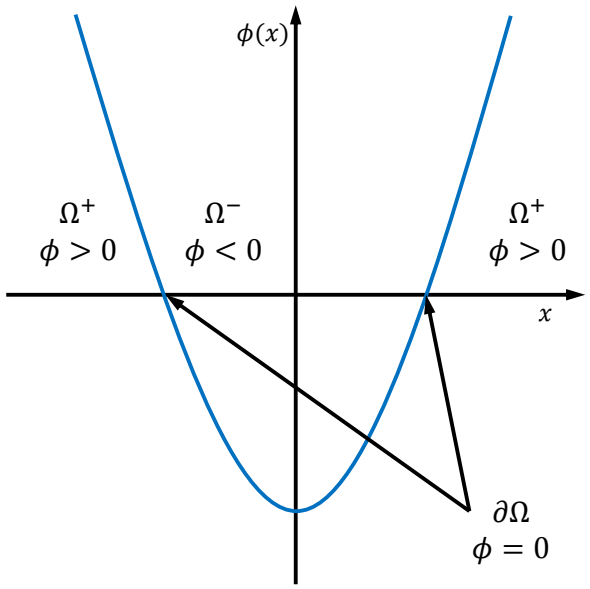


Figure 4.1: The function $\phi = x^2 - 4$ marking the interfaces of three subdomains in 1D

representation of the interface, in real world CFD problems it is unlikely that it is possible to represent a complex deformable interface with an analytical solution.

Computationally, a disadvantage to an implicit representation of interfaces is that an n – dimensional set is required, whereas an explicit representation uses an $(n-1)$ – dimensional set. This means choosing an implicit tracking method can cost more memory and computation time. Authors such as Adalsteinsson and Sethian (1995); Sethian (1996); Peng *et al.* (1999) have all made significant attempts to improve the efficiency of the level set method by confining the calculated signed distance to a narrow band close to the interface. As one moves away from the interface there is no advantage to accurately calculating the level set function and so this can be left unresolved.

Despite implicit interface functions potentially requiring more computational effort, there are some useful geometric properties that can be taken advantage of. The gradient of the function can be calculated by approximating partial derivatives and the result can be used in calculations with the unit normal, perpendicular to the interface, which is immediately calculable. These properties are essential to the reinitialisation methods presented in Sections 4.2 and 4.5. Furthermore the curvature of the interface is obtained easily, being of importance when calculating jump conditions at the free surface of flow solvers such as Watanabe *et al.* (2008) and surface tension calculations like Nourgaliev *et al.* (2005a).

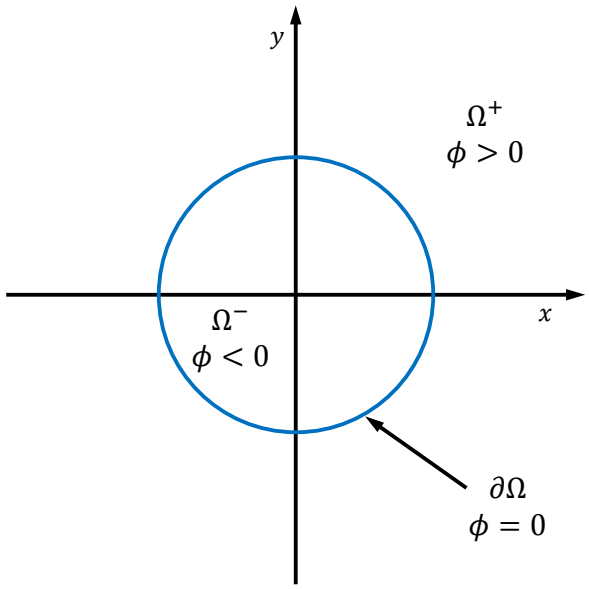


Figure 4.2: The isocontour $\phi = x^2 + y^2 - 1 = 0$ marking the interface of two subdomains

The gradient of a function in 3D is defined as

$$\nabla\phi = \left(\frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y}, \frac{\partial\phi}{\partial z}\right). \tag{4.1}$$

The gradient $\nabla\phi$ runs perpendicular to the isocontours of ϕ and points in the direction of increasing ϕ. Therefore, if $\mathbf{x}_o$ is a point on the zero isocontour of ϕ, i.e. a point on the interface, then $\nabla\phi$ evaluated at $\mathbf{x}_o$ is a vector that points in the same direction as the local outward unit outward normal $\mathbf{N}$ to the interface. The unit normal is

$$\mathbf{N} = \frac{\nabla\phi}{|\nabla\phi|} \tag{4.2}$$

for points on the interface.

The mean curvature of the interface is defined as the divergence of the normal $\mathbf{N} = (n_1, n_2, n_3)$,

$$\kappa = \nabla \cdot \mathbf{N} = \frac{\partial n_1}{\partial x} + \frac{\partial n_2}{\partial y} + \frac{\partial n_3}{\partial z}, \tag{4.3}$$

so that $\kappa > 0$ for convex regions, $\kappa < 0$ for concave regions, and $\kappa = 0$ for a plane. While one could simply use finite differences to compute the derivatives of the components of the normal, it is usually more convenient, compact and accurate to calculate the curvature directly from the

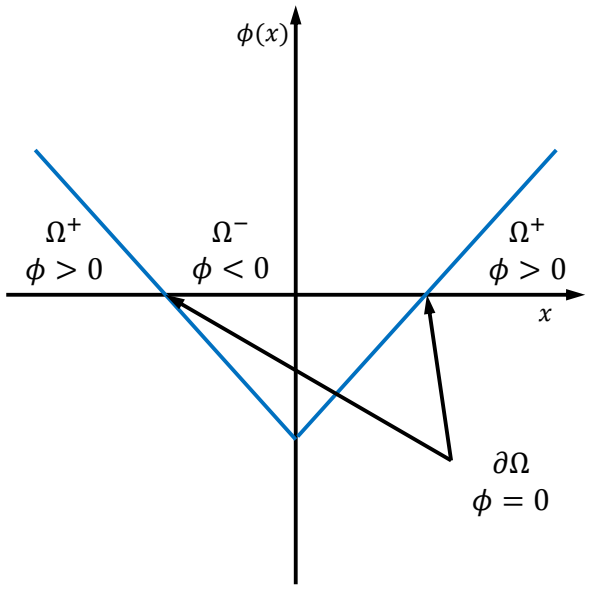


Figure 4.3: A signed distance function marking the interface between three subdomains in 1D

values of ϕ.

$$\kappa = \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|} \right), \tag{4.4}$$

The final, important property that has not been described is that of signed distance. In addition to sharing all the properties of other implicit functions the gradient of the function is constrained as

$$|\nabla \phi| = 1. \tag{4.5}$$

The greatest benefit of using signed distance functions in this manner is they are smooth and monotonic across the interface and can therefore be differentiated accurately and with confidence. The exception to this rule is for a point that is equidistant from at least two points on an interface. At this point, distance functions have a sharp turning point where the gradient is not defined. Figure 4.3 shows the same subdomains as Figure 4.1 this time represented by a signed distance function $\phi(x) = |x| - 2$. For this example, the point $x = 0$ is the point equidistant between the two interface points and some care is required in providing a suitable value for use in simulations.

In the majority of scenarios, the turning point of the distance function will fall between grid points, and does not need to be resolved. The gradient approximation will simply smear the turning point across two grid points. If a turning point does occur directly on a grid point Osher

and Fedkiw (2002, p21) mention that due to it occuring equidistant between two interfaces it is unlikely to impact on the interface itself. However the exception is theoretically possible and in Chapter 5 this issue will be raised, with a special case algorithm discussed and implemented.

As the level set function is transported around the computational domain various phenomena contribute to loss of its signed distance property. As such, the accuracy of interface tracking can be compromised. The solutions that have been successful in improving level set interface tracking have been outlined in Chapter 2. This thesis chooses to focus on the robust and algorithmically simple to implement level set reinitialisation procedure.

One effective and simple method of maintaining the signed distance property of the level set function is to stop the simulation in time, freeze the position of the zero contour of the level set and then redistance the function away from the zero contour position. A naïve implementation of level set re-distancing would be to stop the simulation and explicitly calculate the minimum distance of each point on the numerical grid to the zero contour. The potential for error in the calculation of the interface position and the laborious, computationally expensive nature of this approach makes it extremely unattractive. A better, more efficient approach is the fast marching method of Osher and Sethian (1988), starting at the interface and marching away into both the Ω^+ and Ω^- subdomains. Clever implementation using a binary tree memory structure ensures it is a better approach than the naïve approach but the calculation must be performed for each point on the grid once, entirely re-distancing the domain from scratch.

Level set reinitialisation for flow solvers, first introduced by Sussman *et al.* (1994), is an improvement in accuracy and efficiency of the above methods. Reinitialisation employs an iterative scheme, taking the existing values of the level set function and adjusting it so that $|\nabla\phi| = 1$. As will be shown in Section 4.2, in detail, the reinitialisation is performed using the very simple equation,

$$\phi_t = \text{sign}(\phi_0)(1 - |\nabla\phi|). \tag{4.6}$$

The advantages of this approach are numerous, the interface never need be found before reinitialisation, saving time and maintaining accuracy. Equation 4.6 has the property of re-distancing the level set function, starting from the interface and propagating outwards in the direction of the normal into both Ω^+ and Ω^- subdomains, allowing for the procedure to be stopped after a sufficiently thick band about the interface has been completed. Because the reinitialisation method uses the existing values of the level set function, if the interface moves a small amount between time-steps, there is little alteration in the signed distance property, allowing for perhaps only one reinitialisation iteration per time-step.

In the following section, the theory of the reinitialisation procedure and the numerical algorithm first published by Sussman *et al.* (1994) is presented. While the original paper includes the reinitialisation procedure in the framework of an incompressible flow solver, for the purposes of this thesis, the level set method with reinitialisation is written as a 'plug-in' method for

any immiscible fluid flow solver. The reinitialisation procedure is demonstrated on a variety of static 1D re-distancing problems before being implemented in conjunction with the 1D CIP advection equation. In 2D, tests of two static re-distancing problems are conducted before the reinitialisation scheme is applied to the circular contour advection problem introduced in Chapter 3. For this chapter, area preservation and shape accuracy comparisons are qualitative, a full numerical analysis of the performance of all schemes is the subject of Chapter 6.

In the work of Sussman *et al.* (1998) and Sussman *et al.* (1999) an additional constraint was added to the reinitialisation procedure. The condition was applied to the narrow band directly surrounding the position of the interface. While Equation 4.6 does not affect the position of the zero level set, in numerical computation, finite difference approximations mean that a slight movement is possible, leading to a reduction in accuracy. By requiring that

$$\partial t \int_{\Omega} H(\phi) = 0, \tag{4.7}$$

where $H(\phi)$ is a smoothed approximation of the sign function and Ω is some fixed domain, the area in the band about the interface can be preserved ensuring the zero level set is not disturbed. The discretised algorithm for the improved reinitialisation algorithm is outlined in Section 4.5 and proceeded by 2D tests including a qualitative comparison of reinitialised 2D CIP advection against un-reinitialised 2D CIP advection.

All of the algorithms and tests in this Chapter are performed using a 1^{st} order upwind approximation that is described in Section 4.2. This order of gradient approximation is not sufficient for highly accurate re-distancing. The authors of Sussman *et al.* (1994, 1998, 1999); Peng *et al.* (1999), favour 2^{nd} and 3^{rd} order ENO approximations such as Shu and Osher (1989). In addition to using higher order spatial discretisation for the reinitialisation method. A high order multi-stage time discretisation such as a Runge-Kutta method, Hirsch (2007, p458), can improve the performance of the algorithm. In the algorithms presented here temporal discretisation is not investigated but a high order method would be recommended for flow simulations.

In this chapter the numerical outline for a reinitialisation scheme by Sussman *et al.* (1994) is described in Section 4.2. 1D static re-distancing numerical tests are performed on a selection of cases provided by Sussman *et al.* (1999) before an example of re-distancing during advection is presented in Section 4.3. 2D static and advection tests are performed before conclusions are drawn.

Introduction of a reinitialisation scheme with an area preserving constraint, to better preserve the position of the level set zero contour is described in Section 4.5. While 1D tests are omitted in this case 2D tests are shown with static re-distancing and a comparison of the two reinitialisation methods against and unreintialised advection result are presented before concluding remarks are made.

4.2 An Outline of the Level Set Reinitialisation Method

In this section, reinitialisation theory is presented followed by the discretised algorithm, introduced by Sussman *et al.* (1994). For simplicity, and to allow comparison with other gradient schemes, the algorithm is presented with a 1st order upwind gradient approximation in 2D. Extension to three dimensions is trivial by simply including the third partial derivative in the appropriate calculations.

The level set defines an interface implicitly with the equation

$$\phi_0(\mathbf{x}) = 0. \tag{4.8}$$

When the level set function begins to lose its property of signed distance it is possible to redistance or reinitialise the function while ϕ remains unchanged at the interface. The reinitialisation procedure is achieved by solving the following problem to steady state

$$\phi_t = S(\phi_0)(1 - |\nabla\phi|), \tag{4.9}$$

with the initial condition of

$$\phi(\mathbf{x}, 0) = \phi_0(\mathbf{x}), \tag{4.10}$$

where ϕ_0 is the initial level set function and $S(\phi_0)$ is the sign of the function. When performing a reinitialisation procedure it is useful to smooth the sign function, to improve the level to which the zero contour position is preserved. This is achieved as

$$S_\varepsilon(\phi_0) = \frac{\phi_0}{\sqrt{\phi_0^2 + \varepsilon^2}}. \tag{4.11}$$

Away from the interface, ϕ will converge to $|\nabla\phi| = 1$. Therefore the equation will converge to the actual distance. The algorithm conveniently avoids finding the interface and is efficient to implement numerically.

To numerically evolve Equation 4.9, the equation is first rewritten as a hyperbolic equation,

$$\phi_t + \mathbf{w} \cdot \nabla\phi = S(\phi_0), \tag{4.12}$$

where

$$\mathbf{w} = S(\phi_0)(\nabla\phi / |\nabla\phi|). \tag{4.13}$$

Rearranged, Equation 4.12 takes the familiar form, seen in Section 3.2, of a hyperbolic equation whose characteristics are given by $\mathbf{w}$. The vector $\mathbf{w}$ is a unit normal always pointing outward from the zero level set ($\phi = 0$). In this algorithm the computations advance in pseudo time-steps, in between the time-steps of the real simulation until a convergence criterion is reached.

A discretised version of the algorithm is presented, in two dimensions using the 1st order, upwind gradient approximation method for $\nabla\phi$.

Upwind 1st order approximations of $\partial\phi/\partial x$ and $\partial\phi/\partial y$ are as follows:

$$a \equiv D_x^-(\phi_{i,j}) = (\phi_{i,j} - \phi_{i-1,j})/\Delta x, \tag{4.14}$$

$$b \equiv D_x^+(\phi_{i,j}) = (\phi_{i+1,j} - \phi_{i,j})/\Delta x, \tag{4.15}$$

$$c \equiv D_y^-(\phi_{i,j}) = (\phi_{i,j} - \phi_{i,j-1})/\Delta y, \tag{4.16}$$

$$d \equiv D_y^+(\phi_{i,j}) = (\phi_{i,j+1} - \phi_{i,j})/\Delta y, \tag{4.17}$$

where $D_x^-\phi_{i,j}$ is the standard notation used to denote the upwind gradient approximation in x for $u > 0$ and $D_x^+\phi_{i,j}$ is the upwind gradient approximation in x for $u < 0$. Equivalent logic applies for $D_y^-\phi_{i,j}$ and $D_y^+\phi i, j$.

The discretised version of Equation 4.11 is simply

$$S_\varepsilon(\phi)_{i,j} = \frac{\phi_{i,j}}{\sqrt{\phi_{i,j}^2 + \varepsilon^2}}. \tag{4.18}$$

In the following work, the average grid spacing h has been used as a value for ε.

The gradient magnitude for the iterative equation is calculated as a Godunov Hamiltonian using the upwind gradient approximations

$$G(\phi)_{i,j} = \begin{cases} \sqrt{\max((a^+)^2,(b^-)^2) + \max((c^+)^2,(d^-)^2)} - 1 \\ \text{if} \quad \phi_{i,j}^0 > 0 \\ \sqrt{\max((a^-)^2,(b^+)^2) + \max((c^-)^2,(d^+)^2)} - 1 \\ \text{if} \quad \phi_{i,j}^0 < 0 \\ 0 \quad \text{otherwise} \end{cases}, \tag{4.19}$$

where $a^+ = \max(a,0)$ and $a^- = \min(a,0)$.

Finally, Equation 4.12 is updated using

$$\phi_{i,j}^{N+1} = \phi_{i,j}^N - \Delta t S_\varepsilon(\phi_{i,j}^0) G(\phi_{i,j}^N), \tag{4.20}$$

where N represents the current pseudo time level with $N+1$ being the proceeding pseudo time level. It would be expected that the reinitialisation step would typically be run after each real time-step of a simulation.

4.3 1–Dimensional Tests of Level Set Reinitialisation

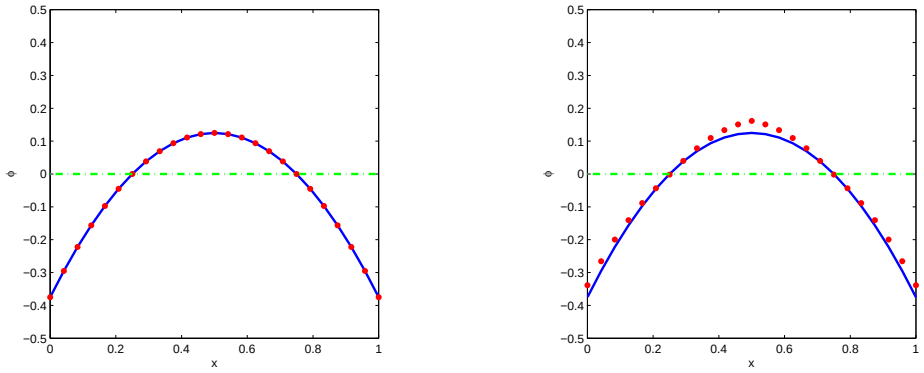
To evaluate the performance of the reinitialisation procedure a series of 1D re-distancing tests are performed. These tests originally appeared in Sussman *et al.* (1999) and demonstrate both static re-distancing and maintaining a signed distance property during advection. The static tests verify that the reinitialisation routine is correctly implemented. The tests are designed to show the functions being re-distanced, without oscillation or other instability. Redistancing should occur starting from the zero points of the function and moving away in both positive and negative subdomains. The location of the zero points should not change significantly during the re-distancing procedure.

For the simple 1D reinitialisation test cases, numerical analysis is not included. Given the simplicity of the re-distancing problem, there is not significant information to be obtained in the error induced due to re-distancing. Likewise, a grid convergence study is not included for these simple problems. The primary function of re-distancing is to remove areas of high curvature as the goal is a function with no curvature and a constant gradient of one. Grid resolution only has a bearing on reinitialisation method's ability to maintain the position of the zero isocontour. A thorough numerical analysis, including grid convergence studies, is presented in Chapter 6 for all of the 2D reinitialisation schemes presented in this thesis.

Figure 4.4 shows the function $\phi_0(x) = -2(x-1/4)(x-3/4)$, re-distanced on a 25 point grid, over 10 time-steps. The test is a similar scenario to re-distancing during simulation in so much as the function is smooth and has a gradient of approximately one in the neighbourhood where the function is zero. The re-distancing algorithm successfully re-distances the function across the whole domain in 10 steps. Of course in a real-world scenario, re-distancing the entirety of the domain is unnecessary, where 2–4 iterations would be sufficient to redistance the locality of the zero points. Furthermore, it can be seen that there is no visible shifting of the position of the zero level set. It was found that a pseudo time-step of $\Delta x/2$ was sufficient to ensure satisfactory re-distancing.

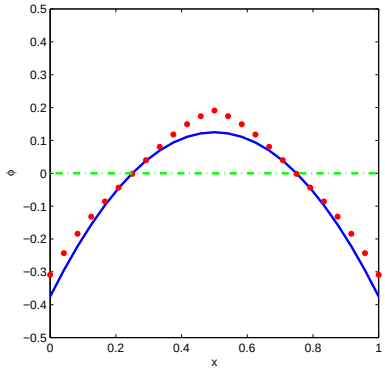
The second test (Figure 4.5) represents the most extreme re-distancing scenario possible. A Heaviside function, whose zero crossings lie somewhere on the discontinuous parts of the function re-distanced. While this situation is unlikely to arise to such an extent during real-world simulation, the reinitialisation scheme must be capable of dealing with locally extreme gradients accurately. The challenge in this test is because the zero isopoints lie in the discontinuous regions and therefore are extremely sensitive to variation in values of ϕ. This causes inaccuracy in the re-distancing approximation to potentially have a more pronounced effect on their positions. Adding to the problem, a large number of iterations is required to redistance the function across the full extent of the domain therefore re-distancing has the potential to create more error in the interface position. The re-distancing is completed successfully over 20 iterations, using a pseudo time-step of $\Delta x/2$, and impressively, the position of the interface at the zero points of the function shows no deviation from its original position.

The final test in this section combines advection, using the CIP 0 method, with the reinitialisation procedure. Reinitialisation is performed once, after each real time-step, for a single iteration with a pseudo time-step of $\Delta x/2$. This is in line with the recommendations made by Sussman *et al.* (1994). Figure 4.6 shows a triangular function (signed distance) advected for 15 cycles of the domain with a CFL number of 0.5. The advection solver manages to preserve the gradient of the triangular function's flat sides however, over numerous time-steps the sharp turning points in Ω^+ and Ω^- lose sharpness due to numerical diffusion over the grid. Figure 4.7 demonstrates the effects of re-distancing each time-step. It is clear that no loss of sharpness occurs with reinitialisation. In the final frame of the simulation, the advected function lies precisely over the initial function with no visible deviation in accuracy.

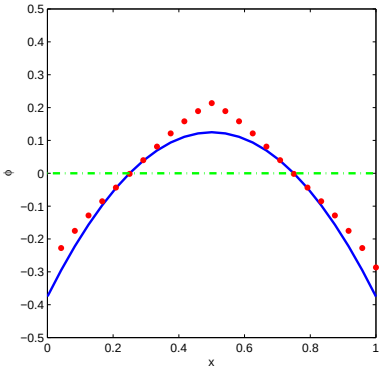


(a) 0 Steps

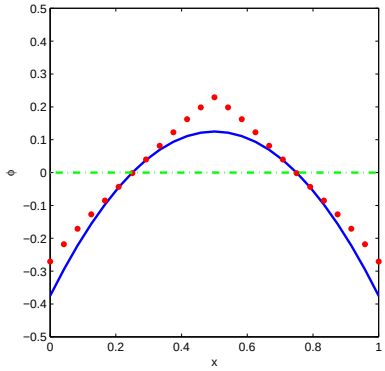
(b) 2 Steps



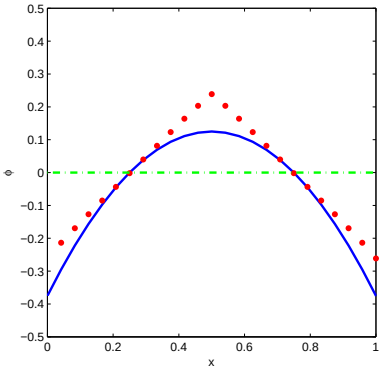
(c) 4 Steps



(d) 6 Steps

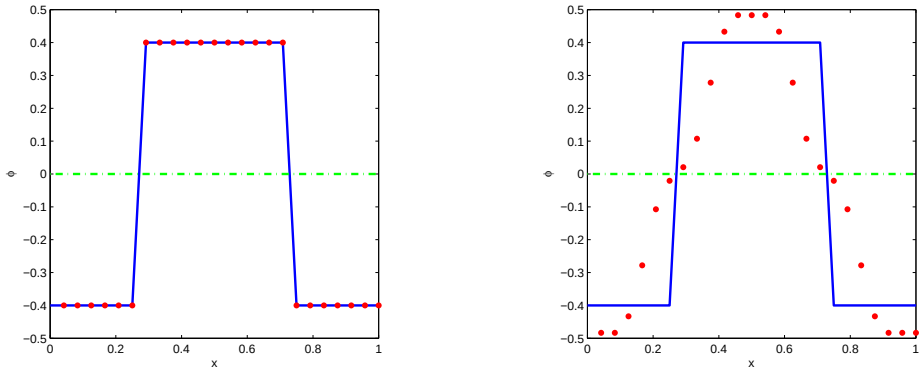


(e) 8 Steps



(f) 10 Steps

Figure 4.4: 1st order reinitialisation of $\phi_0(x) = -2(x-1/4)(x-3/4),\ \Delta t = \Delta x/2$, 25 point grid



(a) 0 Steps

(b) 4 Steps

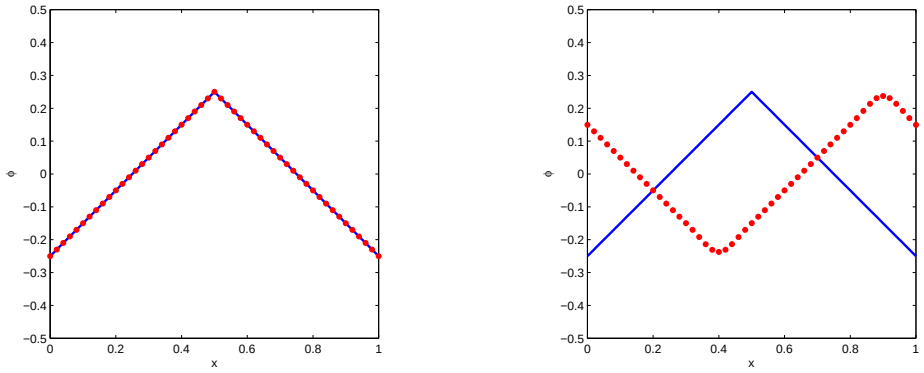
(c) 8 Steps

(d) 12 Steps

(e) 16 Steps

(f) 20 Steps

Figure 4.5: 1st order reinitialisation of a Heaviside function, $\Delta t = \Delta x/2$, 25 point grid



(a) 0% of simulation

(b) 20% of simulation

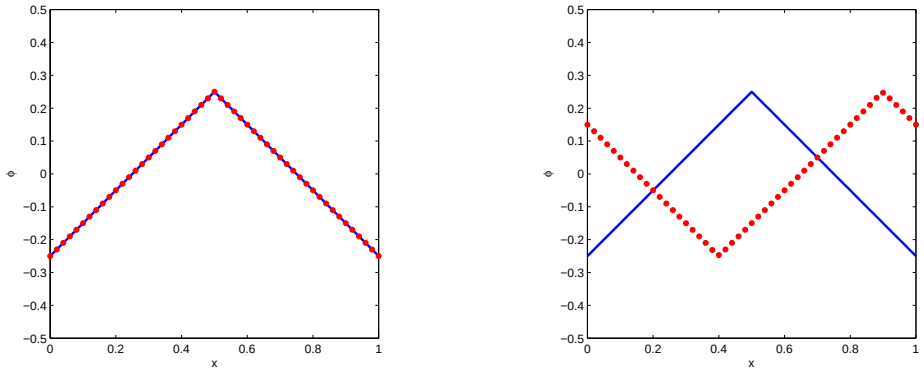
(c) 40% of simulation

(d) 60% of simulation

(e) 80% of simulation

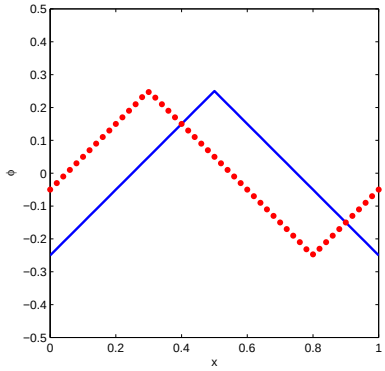
(f) 100% of simulation

Figure 4.6: Triangular wave function advected for 16 cycles with no level set reinitialisation

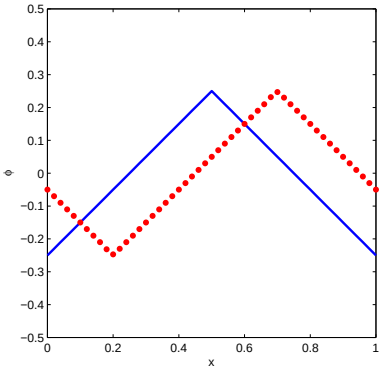


(a) 0% of simulation

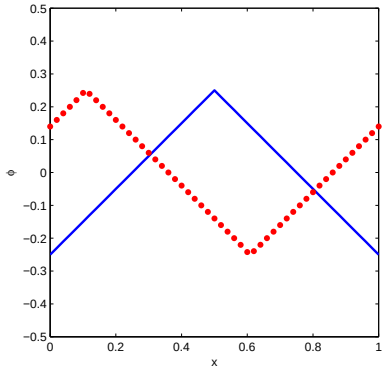
(b) 20% of simulation



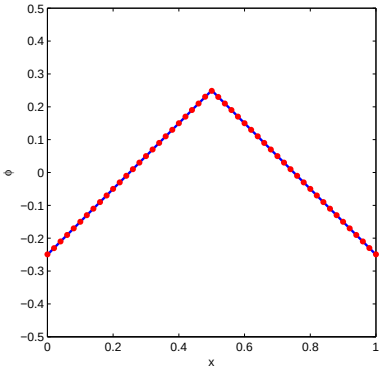
(c) 40% of simulation



(d) 60% of simulation



(e) 80% of simulation



(f) 100% of simulation

Figure 4.7: Triangular wave function advected for 16 cycles with a 1^{st} order accurate level set reinitialisation

4.4 2–Dimensional Tests of Level Set Reinitialisation

Reinitialising a level set function accurately in two dimensions is a greater challenge. The gradient components in both x and y dimensions contribute to the important quantity of gradient magnitude $\nabla\phi$. Inaccuracy of the gradient approximations in either dimension will have a profound effect on the reinitialisation procedure's ability to redistance without disturbing the position of the zero level set. In this section two static re-distancing tests, similar to their 1D counterparts are performed, followed by a comparison of 2D advection with and without reinitialisation.

The first static test requires the surface $\phi = x^2 + y^2 - 1/4$ to be re-distanced. The function creates a circular zero isocontour of radius 0.5 about an origin at the centre of a domain with boundaries of $(-1.0 \leq x \leq 1.0, -1.0 \leq y \leq 1.0)$ discretised to a 101×101 grid. The odd number of points purposefully places the turning point directly on a grid point testing the reinitialisation procedure's ability to deal with a kink in the gradient.

The second static test re-distances a top hat surface (a 2D Heaviside function). A similar circular contour of radius about 0.5 is used on a 101×101 resolution grid. The test examines the reinitialisation method's ability to accurately redistance this extreme, discontinuous function without changing the area bounded by the circular contour. In both tests the zero contour is depicted by a white contour.

	Parablic Surface 200 Iterations	Top Hat Surface 600 Iterations
Initial Area	0.784993	0.781867
Final Area	0.784992	0.779671
Initial Perimeter	3.141007	3.448483
Final Perimeter	3.140361	3.143183

Table 4.1: Contour area and perimeter before and after re-distancing using 1st order reinitialisation method

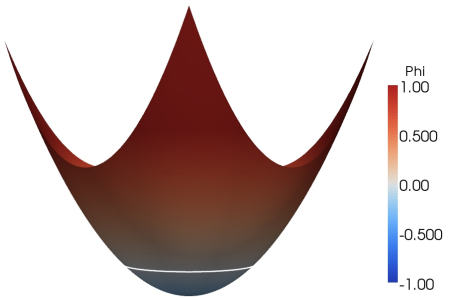
As it is somewhat difficult to evaluate the extent to which the zero isocontour has moved the various values for the contour area and perimeter are summarised in Table 4.1. Redistancing in two dimensions has some effect on the interface. In the case of re-distancing the parabolic surface there is a 0.021% loss of area. This is negligible considering the reinitialisation has been conducted over 200 iterations. However the area loss of the 8.85% in the top hat re-distancing test is more pronounced. There is some movement applied to the interface when reinitialised, however the discontinuous nature of the of the surface makes the area difficult to measure. The area calculating algorithm in Appendix B is not well suited to areas represented by a discontinuous function as it relies on a linear interpolation to estimate the position of the zero contour. When examining the perimeter of the contour it can be seen that a more modest

loss of 0.3% of the original value occurs. As the surface has undergone 600 iterations of the reinitialisation method these results are acceptable. Of equal importance, during both tests the reinitialisation method has shown itself to be numerically stable and coped with the turning point at the centre of the domain.

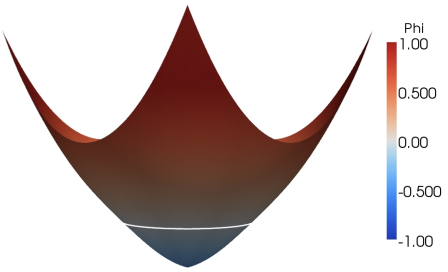
To test the reinitialisation scheme with advection, the circular contour advection problem presented in Chapter 3 is repeated. The simulation is conducted on the 193×193 grid used in the grid convergence study. In the previous chapter it was shown in the grid convergence study, this grid resolution slightly under-resolves the problem. It is an ideal candidate for testing how the reinitialisation procedure affects the accuracy of the interface. After three revolutions of the domain distortion of the contour is evident and any improvements the reinitialisation scheme can make should be visible. The reinitialisation parameters that were employed over the course of the simulation were one reinitialisation iteration after each time-step was completed and a pseudo time-step of $h/10$. These are the optimal values for the parameters as reported by Sussman *et al.* (1994).

The result of the simulation is found in Figure 4.10. The white contour represents the original contour while the black is the un-reinitialised result, using the CIP advection scheme without reinitialisation, and the red contour is produced with reinitialisation. The result of the simulation could be viewed as somewhat surprising, given the promising results in Section 4.3. The figure shows that the contour produced using reinitialisation finishes in the same position as the un-reinitialised result with a similar shape and slightly worse area preservation. This is not a result that promises improvements in interface tracking when the interface is of a more complex topology.

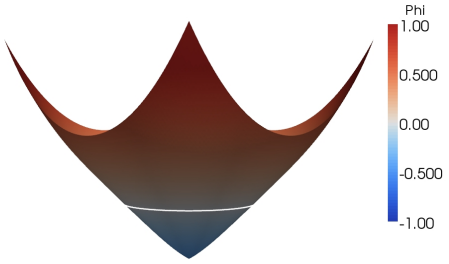
The numerical results for this problem are presented and discussed in Chapter 6 and compared with the subsequent reinitialisation methods. The immediate issue is that of finding a more accurate method of reinitialisation. While a slight decrease in interface tracking accuracy is disappointing, it is not that surprising that a 1^{st} order accurate gradient approximation does not provide the best results possible. Before examining higher order alternatives for the gradient approximation, an area preserving constraint is introduced in an attempt to improve the reinitialisation procedure.



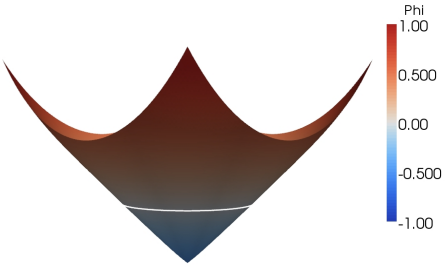
(a) 0 Iterations



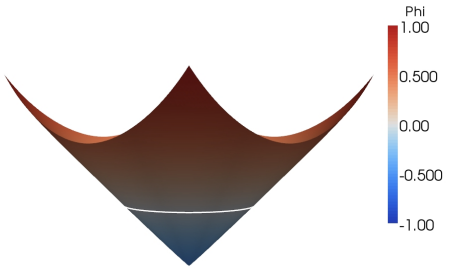
(b) 80 Iterations



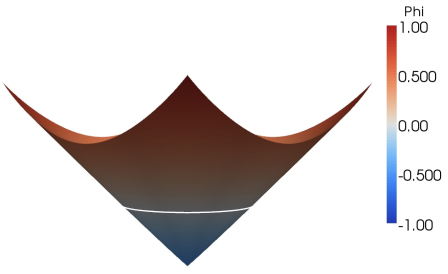
(c) 160 Iterations



(d) 240 Iterations

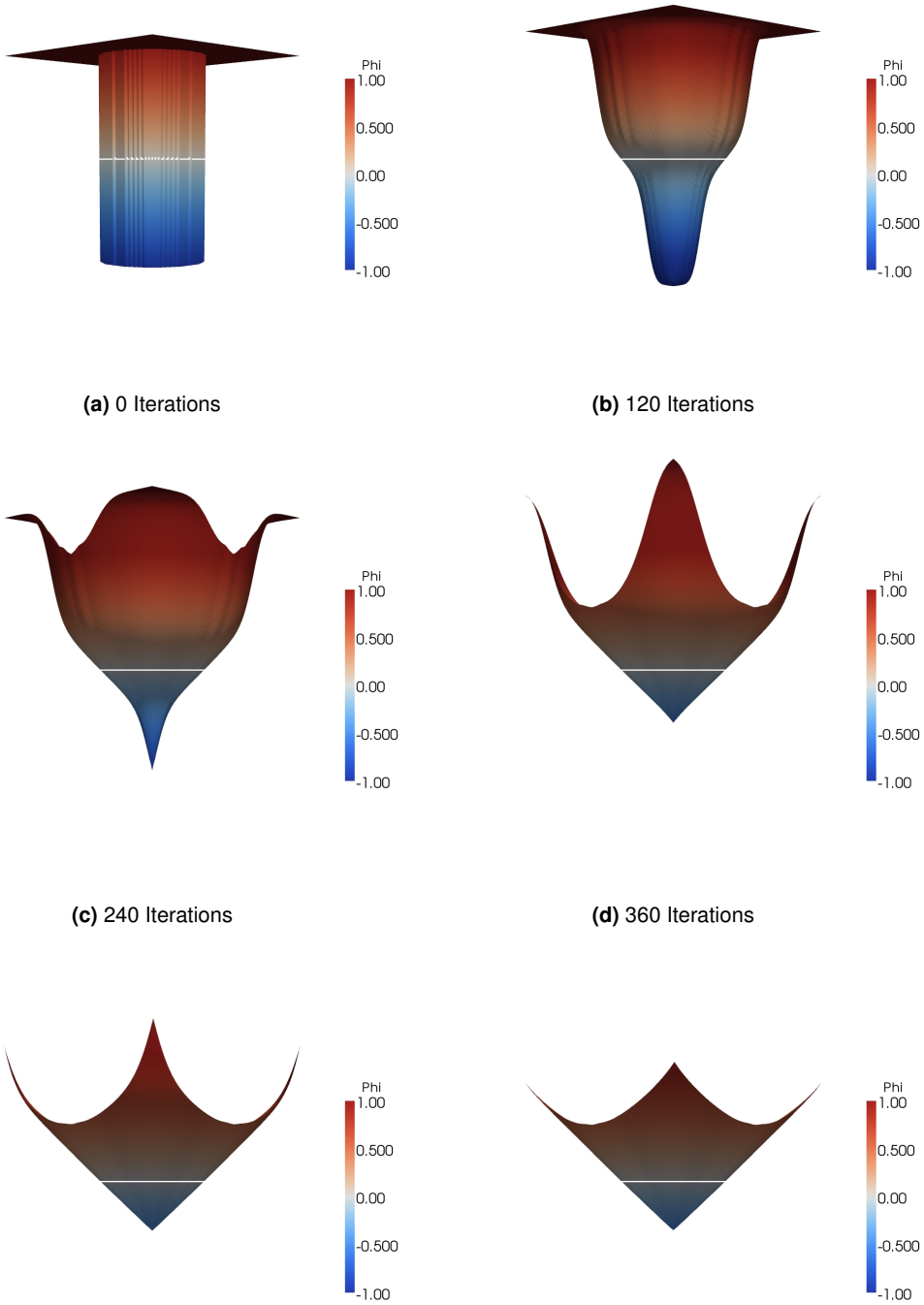


(e) 320 Iterations



(f) 400 Iterations

Figure 4.8: 1st order reinitialisation of $\phi = x^2 + y^2 - 1/4$



(a) 0 Iterations

(b) 120 Iterations

(c) 240 Iterations

(d) 360 Iterations

(e) 480 Iterations

(f) 600 Iterations

Figure 4.9: 1st order reinitialisation of a 2D top hat surface

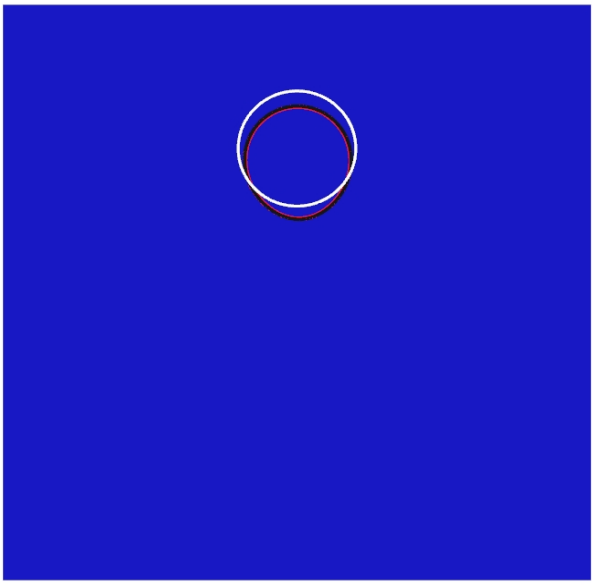


Figure 4.10: Contour advection after 3 rotations. Initial contour (white), un-reinitialised result (black), reinitialised result (red)

4.5 An Improved Interface Preserving Reinitialisation Method

The publication by Sussman *et al.* (1999) recognised that inaccuracy in the gradient approximation procedure during reinitialisation resulted in slight movement of the level set zero contour. This is undesirable, leading to mass loss and poor shape accuracy. In an attempt to improve the reinitialisation routine an area preserving constraint was introduced. The area preserving constraint is calculated in a narrow band about the zero level set, where its value should never change. It is introduced as a modification to the reinitialisation equation as a coefficient in Equation 4.26. This method is outlined below and tested using 2D test problems in Section 4.6.

A step function is defined to represent the signed sets of the level set function. This is a Heaviside function, smoothed at the jump condition to improve the discrete numerics and given by

$$H(\phi_0) \equiv \begin{cases} 1 & \text{if } \phi > \varepsilon, \\ 0 & \text{if } \phi < -\varepsilon, \\ \frac{1}{2}\left(1 + \frac{\phi}{\varepsilon} + \frac{1}{\pi}\sin\left(\frac{\pi\phi}{\varepsilon}\right)\right) & \text{otherwise} \end{cases} \tag{4.21}$$

In a deviation from the original method, in which Sussman *et al.* (1999) assumes equal spacing of $\varepsilon = \Delta x = \Delta y$ the band of values over which the smoothing takes place is broadened such that

$$\varepsilon = \sqrt{\Delta x^2 + \Delta y^2}. \tag{4.22}$$

Having made this change, it was found that Sethian (2001, p15) recommends that ε is treated as a tunable variable that determines the bandwidth of numerical smearing. Typical values should be in the range of $\varepsilon = 1.5\Delta x$, validating what has been shown in Equation 4.22. One advantage of Equation 4.22 could be that, for grids of different resolution in each axis i.e. $\Delta x \neq \Delta y$, a reasonable bandwidth estimate will be produced for all but the most extreme grids.

Equation 4.21 can be used to produce a smoothed sign function for ϕ,

$$S_{\Delta x}(\phi_0) \equiv 2\left(H(\phi_0) - \frac{1}{2}\right). \tag{4.23}$$

The improved reinitialisation method by Sussman *et al.* (1999) requires a similar method of gradient calculation as the previous version presented. Equation 4.19 can be used to produce an upwind gradient estimate of $\nabla\phi^N$, however Equation 4.20 is modified for the gradient projection to

$$\tilde{\phi}^{N+1} = \phi^N + \Delta t L(\phi_0, \phi^N), \tag{4.24}$$

where

$$L(\phi_0, \phi_n) = S(\phi_0)(1 - |\nabla\phi^N|). \tag{4.25}$$

The re-distancing operation of gradient projection is

$$\phi^{N+1} = \tilde{\phi}^{N+1} + \Delta t \lambda_{i,j} H'(\phi_0)|\nabla\phi_0|, \tag{4.26}$$

where

$$\lambda_{i,j} = \frac{-\int_{\Omega_{i,j}} H'(\phi_0)\frac{\tilde{\phi}^{N+1}-\phi_0}{\Delta t}}{\int_{\Omega_{i,j}}[H'(\phi_0)]^2|\nabla\phi_0|}. \tag{4.27}$$

The gradient of the the Heaviside function, which appears in both Equation 4.26 and 4.27 can be calculated as

$$H'(\phi_0) \equiv \begin{cases} 0 & \text{if } \phi > \varepsilon \\ 0 & \text{if } \phi < -\varepsilon \\ \frac{1}{2}\left(\frac{1}{\varepsilon} + \frac{1}{\varepsilon}\cos\left(\frac{\pi\phi}{\varepsilon}\right)\right) & \text{otherwise.} \end{cases} \tag{4.28}$$

The numerical integration over the domain to ensure area conservation

$$\Omega_{ij} = ((x,y)|x_{i-1/2} < x < x_{i+1/2} \text{ and } y_{j-1/2} < y < y_{j+1/2} \tag{4.29}$$

is computed using a nine point stencil

$$\int_{\Omega_{ij}} g \approx \frac{h^2}{24}\left(16g_{ij} + \sum_{m,n=-1;(m,n)\neq(0,0)}^{1} + g_{i+m,j+n}\right). \tag{4.30}$$

For large parts of the computational domain the gradient of the Heaviside function $H'(\phi_0)$ will be zero. This has the effect of cancelling out a large number of terms in Equation 4.26. If this is the case a large saving can be made in computational effort by simple not calculating Equations 4.27 – 4.30, instead taking Equation 4.24 for the gradient projection.

4.6 2–Dimensional Tests of the Improved Reinitialisation Method

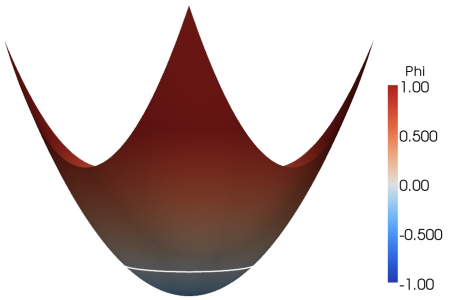
In Sections 4.3 and 4.4 it was shown that the reinitialisation procedure by Sussman *et al.* (1994) was sufficient to accurately redistance static functions and an advected function in 1D. However, in 2D the low order gradient approximation proved insufficient to provide an improvement in shape preservation of a circular interface undergoing advection. Only the 2D tests will be presented in this section, for the sake of brevity. It suffices to say that in 1D the improved reinitialisation (Sussman *et al.*, 1999) scheme did not perform significantly differently to its predecessor.

For the second time, the static reinitialisation tests are presented in Figures 4.11 and 4.12. There are no visual changes to the results, however the numerical results show some significant differences in Table 4.2. Over 200 iterations, reinitialisation of the parabolic surface caused an area loss of just 0.004%, an excellent performance. Conversely, over 600 iterations in the top hat test 10.82% of the area was lost and 4.50% of the perimeter which is significantly more than the simple reinitialisation scheme, clearly the large number of reinitialisation iterations having more of an impact here.

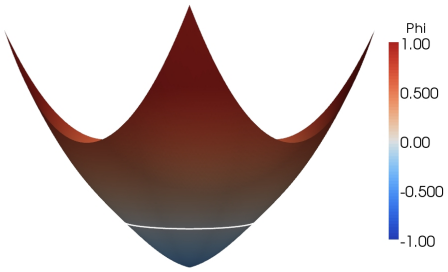
	Parablic Surface 200 Iterations	Top Hat Surface 600 Iterations
Initial Area	0.784993	0.781867
Final Area	0.784889	0.746661
Initial Perimeter	3.141007	3.448483
Final Perimeter	3.140871	3.075367

Table 4.2: Contour area and perimeter before and after re-distancing using improved reinitialisation method

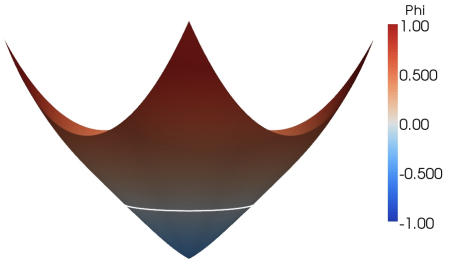
The comparison of advection with and without reinitialisation is included in Figure 4.13. In this figure it can be seen that the improved reinitialisation method has a more pronounced affect on the contour. The results show the shape has been improved significantly over the previous reinitialisation method. However there is again some area loss of the contour area. This is not unexpected, due to the low resolution of the grid but it appears the area loss is greater than that of the unreinitialised result. Given the poor performance of both reinitialisation schemes in 2D thus far, there is a compelling reason to introduce a higher order gradient approximation method as well as the opportunity to adapt the reinitialisation for use on Cartesian cut-cell and other unstructured grids.



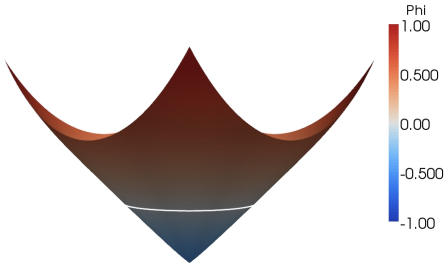
(a) 0 Iterations



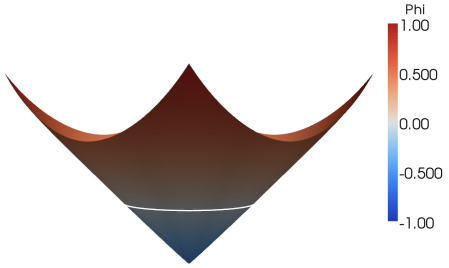
(b) 80 Iterations



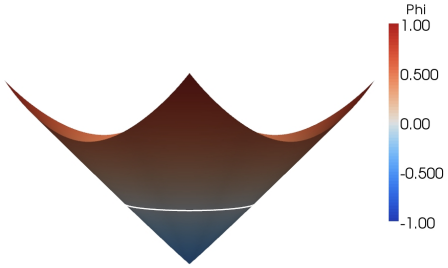
(c) 160 Iterations



(d) 240 Iterations

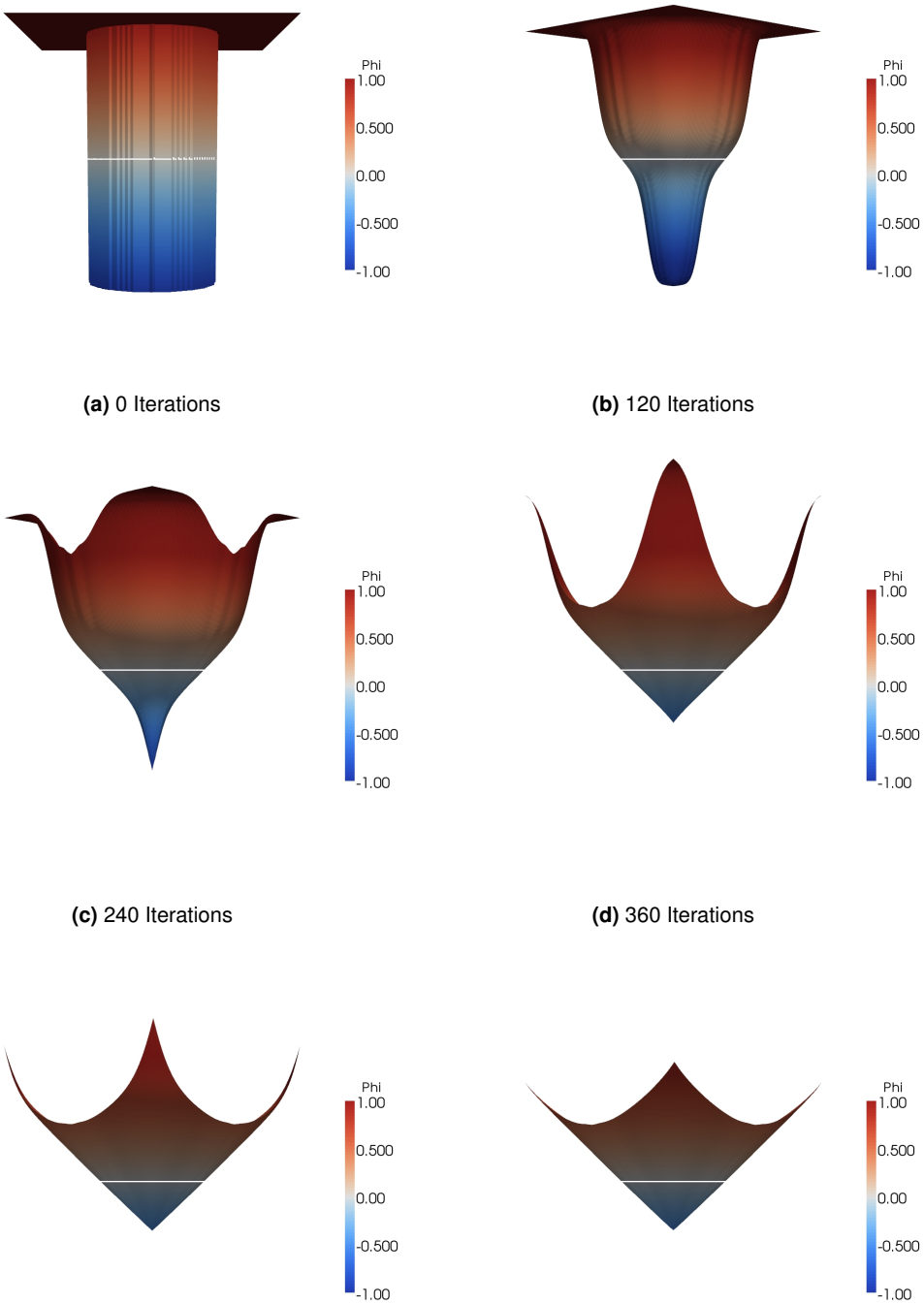


(e) 320 Iterations



(f) 400 Iterations

Figure 4.11: 1st order reinitialisation of $\phi = x^2 + y^2 - 1/4$



(a) 0 Iterations

(b) 120 Iterations

(c) 240 Iterations

(d) 360 Iterations

(e) 480 Iterations

(f) 600 Iterations

Figure 4.12: 1st order reinitialisation of a 2D top hat surface

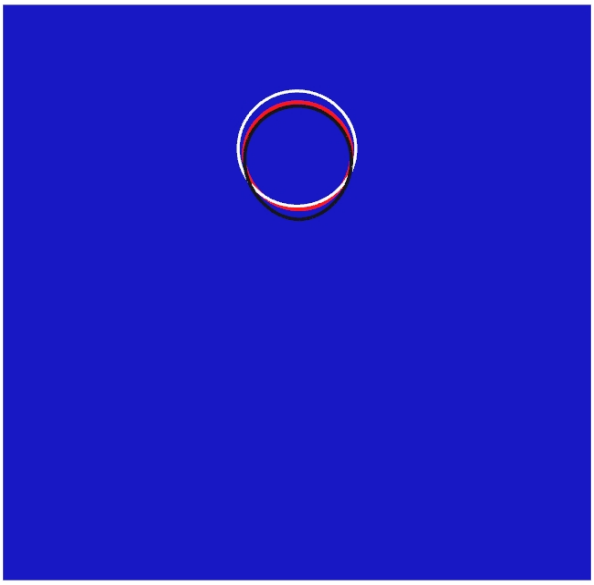


Figure 4.13: Contour advection after 3 rotations. Initial contour (white), un-reinitialised result (black), reinitialised result (red)

4.7 Concluding Remarks on Level Set Reinitialisation

In this chapter a working level set reinitialisation method by Sussman *et al.* (1994) using 1st order upwind gradient approximations has been presented and demonstrated. While this method produces promising results re-distancing functions in a static situation and even 1D advection cases, when the method is implemented in conjunction with the CIP advection solver in 2D there is no visible improvement to the quality of the results by insuring that the level set function is maintained as a signed distance function. The measured results suggest a slight decrease in area preservation performance.

Introducing an area preserving constraint, suggested in Sussman *et al.* (1998, 1999), to improve the performance of the level set reinitialisation method can be seen to maintain a better representation of the contour's shape, however area loss is still a significant factor in these results. The area loss is particularly severe in the 2D top hat re-distancing test due to the high number of reinitialisation iterations and the poor accuracy of the area measurement algorithm at the discontinuous sides of the function.

It is clear that 1st order upwind gradient approximations are not sufficient to yield the accuracy that scientists and engineers expect when using interface tracking methods. The authors of the methods in this chapter have recommended use of an ENO gradient approximation scheme (Shu and Osher, 1989) in their reinitialisation routines. In the following chapter, a new higher order gradient approximation will be introduced that will improve the performance of the level set reinitialisation method and prepare it for extension onto a mixed geometry, unstructured grids such as a Cartesian cut-cell grid.

Chapter 5

A Limited Least Squares Scheme for Level Set Reinitialisation

In the previous chapter two reinitialisation methods were introduced and tested using 1^{st} order accurate gradient approximations. While the monotonic nature of 1^{st} order upwind approximations is essential for the stability of level set re-distancing, their order of accuracy was insufficient to enable reinitialisation to provide significantly improved interface tracking. In spite of implementing a constraint, designed to minimise movement of the zero level set isocontour during reinitialisation, there was a degree of area loss from the circular contour in the tests performed. In an attempt to improve the situation, this chapter will introduce a substantial step towards a reinitialisation procedure with a novel gradient approximation method, suitable for use with mixed geometry unstructured grids.

ENO schemes, are already used extensively in level set reinitialisation, though are not necessarily the best approach. Authors Sussman *et al.* (1994, 1998, 1999); Peng *et al.* (1999); Yue *et al.* (2003) all use ENO schemes by Shu and Osher (1989); Jiang and Shu (1996) for the solution of the reinitialisation equation. One of the principle advantages of these schemes is the level set is treated as an integral part of the incompressible flow solver. The ENO scheme is used to evaluate fluxes throughout the algorithm, making efficient use of calculated data and computer code. It makes good sense to make use of this method utilised when solving level set equations, first demonstrated by Osher and Sethian (1988). To extend these methods, improving accuracy, the commonly used gradient approximation is a higher order ENO scheme such as Shu and Osher (1989). To do so presents a challenge on an unstructured grid due to their use of divided difference tables leading to a complex implementation.

While this is advantageous in schemes where the level set function is treated as an incorporated variable there are also cases where the advantage is diminished. If the incompressible flow solver by Watanabe and Saeki (2002) is taken as an example, it is desirable to calculate the movement of the level set function independently from resolving the flow field. Mulder and Osher (1992) showed that in the case of wave problems with a smooth velocity and severe discontinuity in the density of the fluids, this approach yielded a more accurate result. Furthermore, the CIP method used to solve the advection phase produces an estimation of function

gradient every time-step as a requirement and a central difference scheme is used to solve the non-advection phase of the flow regime. Therefore there is no requirement for an ENO approximation method. The finite volume CIP method by Ii and Xiao (2007) is a suitable future candidate for implementation on a Cartesian cut-cell grid. This method makes use of a slope limiter to produce a TVB approximation for the advection phase, consequently here too, an ENO approximation scheme cannot be shared.

A large number of ENO schemes have been developed for use on uniform Cartesian grids. ENO and WENO schemes have also been extended to adaptive meshes (Nourgaliev *et al.*, 2006; Min and Gibou, 2007) with some success. However spatially adaptive methods generally still consist of regularly spaced rectangular (2D) or hexahedral (3D) cells. There are significantly fewer high resolution upwind schemes available for use on unstructured grids. One of the principle reasons for this is the difficulty of obtaining data lying directly along an axis, at some appropriately spaced interval, with which to determine some monotonicity criterion. ENO methods tend to use a stencil of points either side of the point currently being calculated. A measure of smoothness is determined using divided differences or similar. This method is impossible to implement if the necessary stencil is not available. Therefore another approach must be found. Of the ENO schemes for unstructured grids that have been published, some of the most widely referred to are those of Barth and Sethian (1998); Zhang and Shu (2003, 2009).

While these ENO schemes successfully implement solutions on unstructured grids there are aspects that make these less than ideal options. The method by Barth and Sethian (1998) while capable of working on both 2D and 3D grids was highly influenced in accuracy by the quality of grid used. In order to maintain the smoothness of the solution an edge flipping method was recommended. Of course the requirement for this is that edges are easily flipped on the grid method that is being used. The 2D WENO method by Zhang and Shu (2003) is a more accurate method but is complicated to implement, using a large grid stencil to produce results. This method was extended to 3D in Zhang and Shu (2009) and the stencil minimised in size. Zaspel and Griebel (2010) reported that implementing this WENO scheme for their GPGPU compute algorithm was problematic due to the complexity of the routine. Finally, the principle limitation of these ENO methods is that they are developed for use on triangular (2D) and tetrahedral (3D) grids, not immediately suited for use on a Cartesian cut-cell grid such as Qian *et al.* (2003). While it may be possible to extend an existing scheme to a mixed geometry grid, it is desirable to seek a simplified routine.

5.1 Developing a Scheme for Level Set Reinitialisation for Unstructured grids

In order to implement a reinitialisation method on a Cartesian cut-cell grid the gradient approximation stage of the reinitialisation method must be compatible. Instead of choosing an ENO method this chapter introduces a new slope limited gradient method, inspired by the ideas of Darwish and Moukalled (2003). The MUSCL TVD method in Darwish and Moukalled (2003) has already been used successfully with Cartesian cut-cell grids. For the first time, a new higher than 1st order monotonic approximation is presented. With minor further development, the scheme is capable of operation on structured, Cartesian cut-cell or other unstructured grid types. It is capable of operating in two or three dimensions with a trivial increase in complexity required to account for the additional dimension.

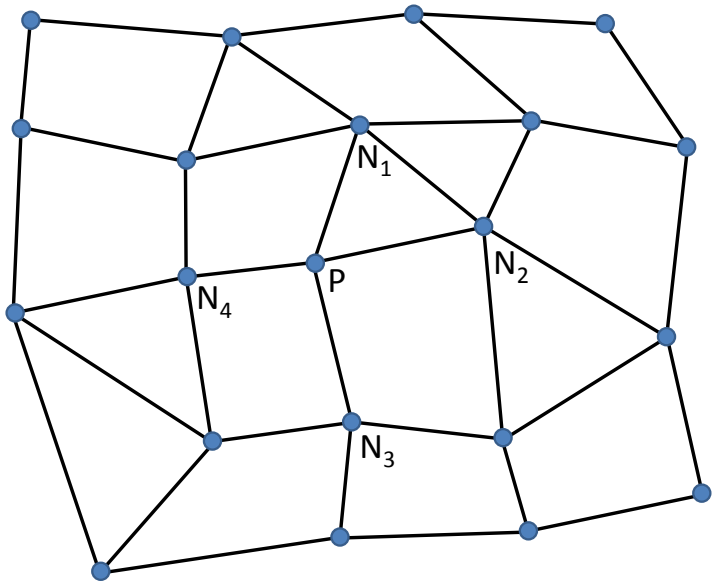


Figure 5.1: An example unstructured grid featuring triangular and quadrilateral cells

In the following sections a generalised example of an unstructured numerical grid of mixed cell geometry will be considered. An example section of such a grid is included in Figure 5.1. One of the inherent complexities of a Cartesian cut-cell grid is any numerical scheme must be flexible enough to process both triangular and quadrilateral cells. The example grid reflects this. If a point, P, is chosen, it is found to have a number of neighbour points that it is connected to, $N_1 \ldots N_n$. In many unstructured and Cartesian cut-cell grids the number of neighbour nodes can vary and, of course, their position in relation to point P is not strictly controlled.

The basis of the new gradient approximation is a least squares statistical approximation with a slope limiting function applied to the approximation in areas where it is required to maintain stable reinitialisation. The approximation has no formal level of accuracy but could be thought

of as a central difference approximation. As Anderson (1995, p497-498) discusses, despite the least squares taking a central bias to differencing and occasionally using points from outside of the zone of influence for a given point, it can work quite successfully. This is especially true for a smooth function such as the level set function. However, level set reinitialisation starts at the zero contour and propagates outward like advection. Advection solutions ideally require an upwind gradient approximation so the use slope limiter is required to guarantee a stable smooth function. While overestimates of gradient result in spurious oscillation and loss of stability the underestimates do not. Tests in this and the following chapters demonstrate that any inaccuracy is minor and the new reinitialisation scheme significantly outperforms 1^{st} order schemes.

In the following sections the least squares method and a new slope limiter, required for performing a stable reinitialisation of the level set function, are introduced (Sections 5.2 and 5.3). For clarity and ease of comparison these methods are outlined for use on a structured grid. Static re-distancing tests are performed in order to assess the performance characteristics of the new method.

An issue associated with the use of a central biased gradient scheme is highlighted by the static re-distancing tests. The appearance of reinitialisation 'spikes' at points where the gradient is indeterminate. In Section 5.5 several approaches to limiting the appearance of unwanted numerical artefacts, in the form of 'reinitialisation spikes', are explored.

The final part of the chapter focuses on the approach that could be taken when extending the scheme to unstructured grids. It requires the reconstruction of the level set values to points that lie directly in the path of the normal of the level set for each grid point. A routine for determining the upwind direction for establishing the reconstructed points is presented.

5.2 The Least Squares Gradient Approximation

The least squares method is a statistical procedure for finding the best-fitting curve to a set of points by minimizing the sum of the squares of their offsets. This procedure can be used in a similar manner to estimate the components of gradient using the point values of a function on a numerical grid. Because this method uses the offset of the points, there is no limitation to the structure of the grid it can be used on, so long as the offset is calculable. It is often thought of as a 2nd order accurate method, although it has no formal level of accuracy in three dimensions. The following method was presented by Sun (1998) in 2D but can be extended to 3D simply by considering the third dimensional component.

The gradient $\nabla\phi = (D_x\phi, D_y\phi)$ is desired at point i and its neighbouring points j.

The gradient is estimated, minimising the distance between ϕ_j and a piecewise approximation

$$\phi'_j = \phi_i + \nabla\phi \cdot \Delta\mathbf{r}_{ji},$$

where $\Delta\mathbf{r}_{ji} = (\Delta x_{ji}, \Delta y_{ji}) = (x_j - x_i, y_j - y_i)$. The distance is expressed as

$$\sum_j (\phi_j - \phi'_j)^2 = \sum_j [\phi_j - (\phi_i + \nabla\phi \cdot \Delta\mathbf{r}_{ji})]^2,$$

where $\sum_j$ represents the summing of all neighbouring points. The resultant linear system is

$$\left(\sum_j \Delta x_{ji}\Delta x_{ji}\right)\phi_x + \left(\sum_j \Delta x_{ji}\Delta y_{ji}\right)\phi_y = \sum_j \Delta x_{ji}\Delta\phi_{ji}$$

$$\left(\sum_j \Delta y_{ji}\Delta x_{ji}\right)\phi_x + \left(\sum_j \Delta y_{ji}\Delta y_{ji}\right)\phi_y = \sum_j \Delta y_{ji}\Delta\phi_{ji},$$

or simply

$$\mathbf{A}\,\mathbf{x} = \mathbf{b},$$

where $\mathbf{A}$ is the matrix of coefficients and $\mathbf{x}$ is $(D_x\phi, D_y\phi)^{\mathrm{T}}$. It follows that

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b},$$

where matrix $\mathbf{A}^{-1}$ is the inverse matrix of $\mathbf{A}$ and its elements satisfy

$$(\mathbf{A}^{-1})_{ij} = \frac{\text{cofactor of } \mathbf{A}_{ji}}{|\mathbf{A}|}.$$

with

$$\mathbf{A}^{-1} = \frac{1}{|\mathbf{A}|} \begin{pmatrix} \sum_j \Delta y_{ji}\Delta y_{ji} & -\sum_j \Delta x_{ji}\Delta y_{ji} \\ -\sum_j \Delta x_{ji}\Delta y_{ji} & \sum_j \Delta x_{ji}\Delta x_{ji} \end{pmatrix}$$

and determinant $|\mathbf{A}| = \left(\sum_j \Delta x_{ji}\Delta x_{ji}\right) \cdot \left(\sum_j \Delta y_{ji}\Delta y_{ji}\right) - \left(\sum_j \Delta x_{ji}\Delta y_{ji}\right)^2$.

Rearranging will allow the gradient components to be calculated as

$$D_x\phi = \frac{\left(\sum_j \Delta y_{ji}\Delta y_{ji}\right)\left(\sum_j \Delta x_{ji}\Delta\phi_{ji}\right) - \left(\sum_j \Delta x_{ji}\Delta y_{ji}\right)\left(\sum_j \Delta y_{ji}\Delta\phi_{ji}\right)}{\left(\sum_j \Delta x_{ji}\Delta x_{ji}\right)\left(\sum_j \Delta y_{ji}\Delta y_{ji}\right) - \left(\sum_j \Delta x_{ji}\Delta y_{ji}\right)^2} \tag{5.1}$$

$$D_y\phi = \frac{\left(\sum_j \Delta x_{ji}\Delta x_{ji}\right)\left(\sum_j \Delta y_{ji}\Delta\phi_{ji}\right) - \left(\sum_j \Delta x_{ji}\Delta y_{ji}\right)\left(\sum_j \Delta x_{ji}\Delta\phi_{ji}\right)}{\left(\sum_j \Delta x_{ji}\Delta x_{ji}\right)\left(\sum_j \Delta y_{ji}\Delta y_{ji}\right) - \left(\sum_j \Delta x_{ji}\Delta y_{ji}\right)^2} \tag{5.2}$$

While this method of gradient approximation can be used on any structure of numerical grid, the estimate uses function values from all directions about the point being evaluated. Previously mentioned, this leads to the inclusion of points in both upwind and downwind directions, causing the potential for instability. In the next section a slope limiting procedure is outlined for guaranteeing stability when using this gradient approximation during reinitialisation.

5.3 A Slope Limited Scheme for Level Set Reinitialisation

The limiting schemes for unstructured grids presented by Darwish and Moukalled (2003), who in turn uses the ideas of Barth and Jesperson (1989), is an inspiration for this new limiting scheme. The method by Darwish and Moukalled (2003) is written for the case of a finite volume solver whose cell data is stored at the cell centres with values requiring polynomial reconstruction at the cell faces. While the application and method has changed significantly, the core difficulty in implementing a slope limited scheme on an unstructured grid remains. Unstructured grids require a virtual upwind node with which to base a slope limiter and it is this issue that is a key influence to the approach detailed in this section. Without a code implementation of a suitable unstructured grid it is impossible to take this to its fullest conclusion. The proof of concept is presented for a structured grid with a description of a possible method of extension to an unstructured grid. The description of the limiter for implementation on a regular Cartesian grid follows.

Once a least squares gradient approximation has been obtained for ϕ it must be limited to the 1st order to provide a stable estimate for use in the reinitialisation procedure. First a limiting coefficient is calculated based on the appropriate first order gradient for the upwind and downwind direction in each dimension. For the two dimensional cases that have been dealt with in this Thesis this will be a North, South, East and West coefficient. The East coefficient is calculated as follows

$$\psi_{i,j}^{\text{East}} = \begin{cases} \frac{\max(\phi_{i-1,j},\phi_{i+1,j})-\phi_{i,j}}{\nabla\phi_{i,j}\cdot\Delta x} & \text{if} \quad \phi_{i+1,j} > \phi_i \\ \frac{\phi_{i,j}-\min(\phi_{i-1,j},\phi_{i+1,j})}{\nabla\phi_{i,j}\cdot\Delta x} & \text{if} \quad \phi_{i+1,j} < \phi_i \\ 1 & \text{otherwise.} \end{cases} \tag{5.3}$$

Repeat this procedure for the West, North and South directions, altering the subscripts as appropriate.

In order to provide stability during the reinitialisation process the limiter coefficient is used to adjust the least squares gradient approximation. If the limiter coefficient is used to adjust the gradient at all points a 1st order gradient estimate is produced. As such, a minimum between the limiter and a unity coefficient is used to prevent the limiter coefficient from being applied to gradient measurements that do not exceed the 1st order estimate.

$$\begin{aligned} D\phi_x^+ &= \nabla\phi \cdot \mathbf{min}(\psi_{i,j}^{East}, 1) \\ D\phi_x^- &= \nabla\phi \cdot \mathbf{min}(\psi_{i,j}^{West}, 1). \end{aligned} \tag{5.4}$$

As a upwind limited gradient has been produced the reinitialisation procedure can be completed in the manner described in Chapter 4. Calculating the Godunov Hamiltonian is completed in the following manner

$$G(\phi)_{i,j} = \begin{cases} \sqrt{\max((a^+)^2,(b^-)^2)+\max((c^+)^2,(d^-)^2)}-1 \\ \text{if} \quad \phi^0_{i,j} > 0 \\ \sqrt{\max((a^-)^2,(b^+)^2)+\max((c^-)^2,(d^+)^2)}-1 \\ \text{if} \quad \phi^0_{i,j} < 0 \\ 0 \quad \text{otherwise} \end{cases}, \tag{5.5}$$

where $a \equiv D_x^{East}$, $b \equiv D_x^{West}$, $c \equiv D_y^{North}$ and $d \equiv D_y^{South}$, and the level set function is re-distanced using

$$\phi^{N+1}_{i,j} = \phi^N_{i,j} - \Delta t S_\varepsilon(\phi^0_{i,j}) G(\phi^N_{i,j}). \tag{5.6}$$

It is also possible to use this method for the gradient approximations in the level set reinitialisation method with the area conserving constraint presented in Section 4.5. This concludes the mathematics of the gradient approximation.

5.4 Static Redistancing Tests

In this section the static re-distancing of the smooth and discontinuous surfaces introduced in the previous chapter, Chapter 4, are presented using the new least squares limited (LSL) reinitialisation method. An interesting phenomenon that appears as a result of using the least squares gradient approximation in the reinitialisation procedure is a spike artefact at the centre of numerical domain, in the location of the minimum. The phenomenon and a solution to it is discussed in Section 5.5. A new test shape, a slotted disc, known as Zalesak's disc (Zalesak, 1979) is introduced for the tests in this section.

The static reinitialisation tests were run until the full domain was re-distanced and a comparison with the results from Chapter 4 could be made. In the case of the parabolic surface test, the test was run to completion until the full domain was re-distanced, reaching a steady state. This took 200 re-distancing steps and the re-distancing sequence is displayed in Figure 5.2. The top hat re-distancing test completed after 600 re-distancing steps and is presented in Figure 5.3. Let it be reiterated that during an unsteady simulation the level set function would only undergo a few iterations of the reinitialisation procedure between any two time-steps. These static re-distancing procedures are designed to represent the more extreme re-distancing scenarios, however they may be of value if the user wishes to initialise a smooth function in a domain before simulation begins.

	Parabolic Surface 200 Iterations	Top Hat Surface 600 Iterations	Zalesak's Disc Surface 100 Iterations
Initial Area	0.784993	0.781867	0.208000
Final FOU Area	0.784992	0.779671	0.207401
Final LSL Area	0.784993	0.779849	0.207581
Initial Perimeter	3.141007	3.448483	2.886351
Final FOU Perimeter	3.140361	3.143183	2.715663
Final LSL Perimeter	3.140877	3.146364	2.738390

Table 5.1: Table of results for least squares limited reinitialisation tests. Comparison with 1^{st} order reinitialisation tests

Similarly to the previous chapter, the static reinitialisation tests do not provide definitive information about the accuracy of the schemes during advection; they serve as an important verification test and allow for comparison with the 1^{st} order results. A short series of numerical results is presented in Table 5.1. Initial area and the perimeter of the initial zero level set are compared with final results for each test and conclusions about the accuracy of the reinitialisation procedure can be made. Due to the static nature of the tests, numerical data is never moved to an adjacent point. There is no deterioration of the contour shapes as a result of numerical errors through diffusion or dispersion due an advection process. It is therefore already expected

that results should be very close to the initial values.

The general trend of area loss during reinitialisation inside the zero contour is continued in these new results. The total area lost in the tests ranged from $< 6 \times 10^{-5}\%$, for the LSL reinitialisation of the parabolic surface, to 0.2876%, during the reinitialisation of Zalesak's Disc, using 1st order upwind approximations. In all situations the LSL reinitialisation scheme conserves area better than the 1st order scheme.

The perimeter distance of the zero level set follows the same trends found when comparing areas. The range of results varies from 0.0041% lost, for the LSL reinitialisation of the parabolic surface, to 5.91% when a 1st order gradient is used in the Zalesak's Disc test. The perimeter values suffer greater losses in the two test cases with discontinuous functions whereas the smooth surface of the parabolic case is re-distanced more accurately. The greater number of reinitialisation iterations and the discontinuous representation of the interface both contribute to this.

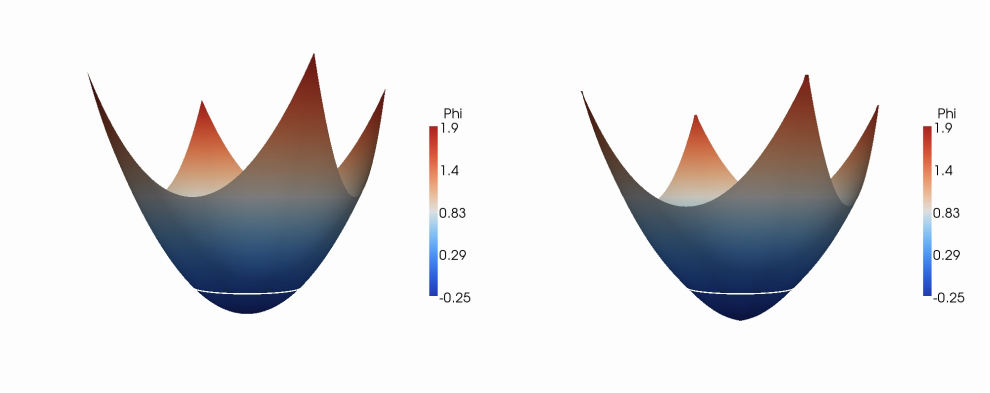
In these test results, the most striking visible feature is numerical artefacting in the form of spikes as each surface is re-distanced. These 'reinitialisation spikes' appear, to some extent, in all of the tests using the LSL gradient approximation. While this phenomenon is unintended and undesirable its effect does not stop the level set function from being re-distanced accurately in a band about the zero level set contour. The cause of the spikes is easily understood. Providing an elegant solution to their appearance is more awkward and is discussed in Section 5.5.

The spikes arise from the reinitialisation formula re-distancing the level set function until its gradient magnitude equals 1 (shown in Equation 5.7). At this point the problem becomes steady state and the function is no longer re-distanced.

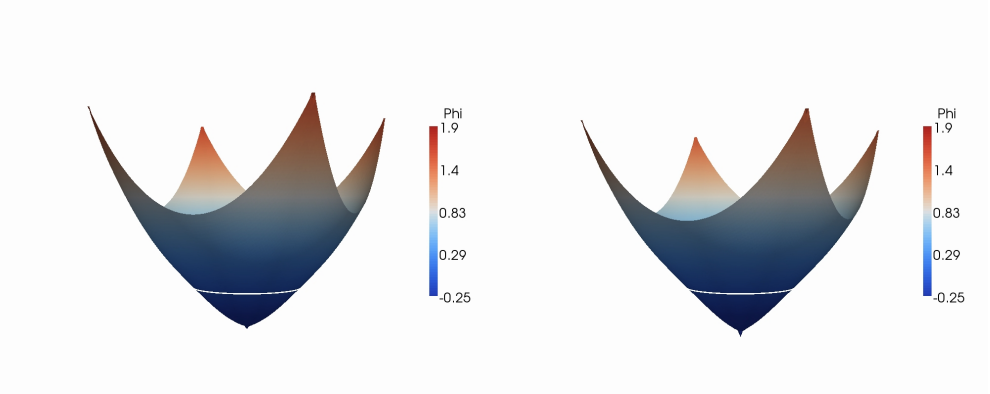
$$\phi_{i,j}^{n+1} = \phi_{i,j}^{n} + \Delta t S_{\varepsilon}(\phi_{i,j}^{0})(1 - G(\phi_{i,j}^{n})) \tag{5.7}$$

The artefacting is caused when a special case scenario is considered. If a surface maxima or minima occurs at point, P, where surrounding points, $N_1 \ldots N_i$, all have similar values to each other. The resultant gradient by the least squares method, or other central based gradient approximation, will be a gradient of or close to zero. This scenario occurs on the 101×101 grid of the parabolic surface and top hat surface tests because the odd number of points cause the function minimum to occur precisely on the central point of the grid.

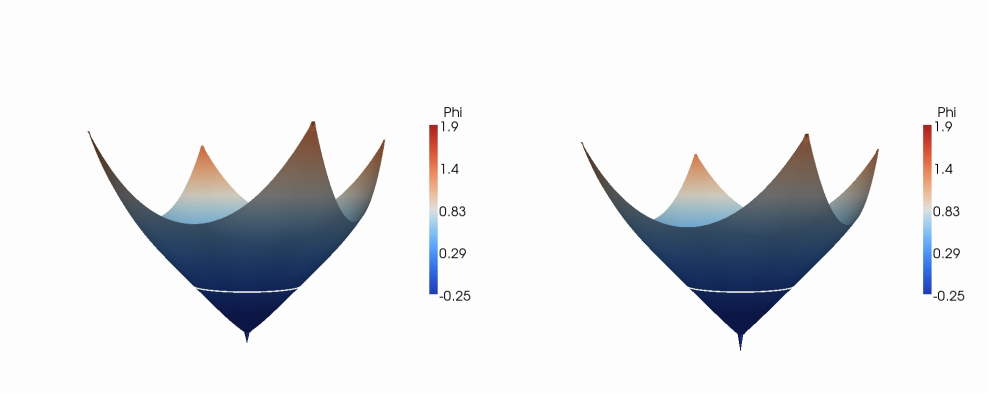
Because of the symmetrical nature of the function on the grid, all the neighbour points have the same value. The resultant gradient calculated at the central point will always remain zero. Technically, the curvature of the minimum at the central grid point is infinite and therefore its gradient is indeterminate. The resultant effect is Equation 5.7 will continue to move the minimum away from the zero level set, creating a spike that will grow indefinitely. Any slope limiting calculation applied to the least squares approximation will not alter the result, the



(a) 0 Steps **(b)** 40 Steps

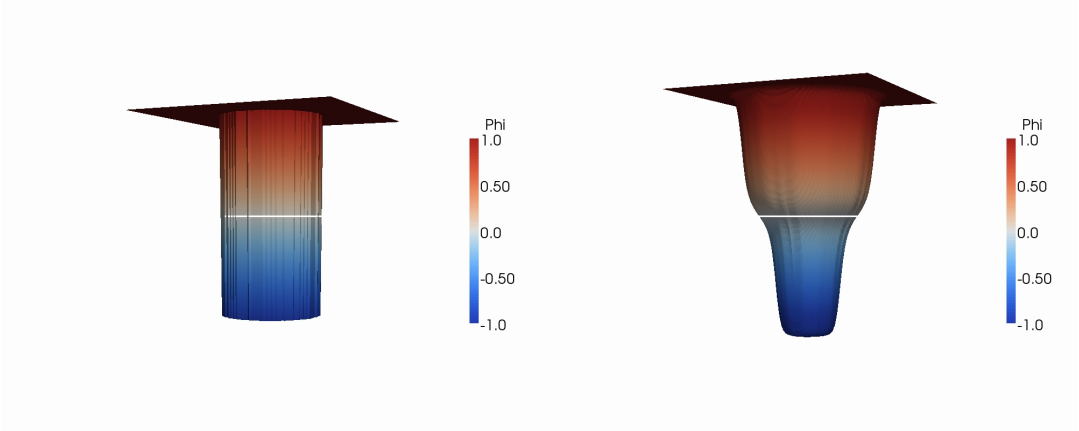


(c) 80 Steps **(d)** 120 Steps



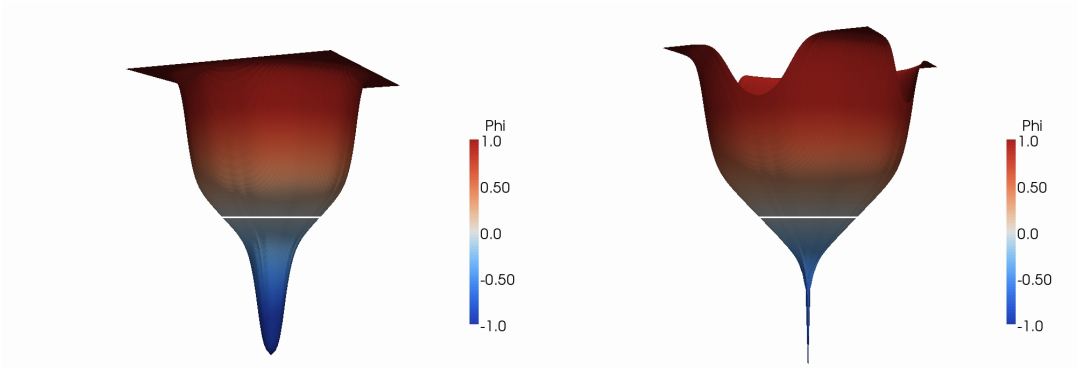
(e) 160 Steps **(f)** 200 Steps

Figure 5.2: The re-distancing of a parabolic surface on a 101×101 resolution grid



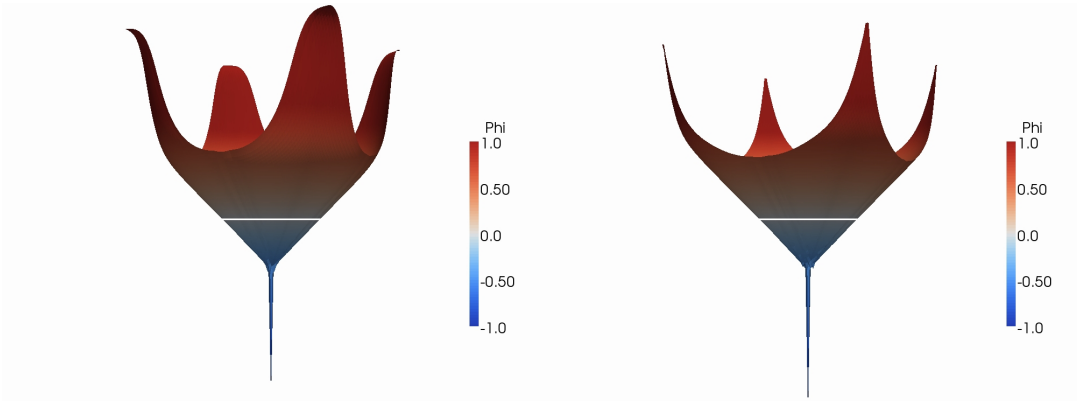
(a) 0 Steps

(b) 100 Steps



(c) 200 Steps

(d) 300 Steps



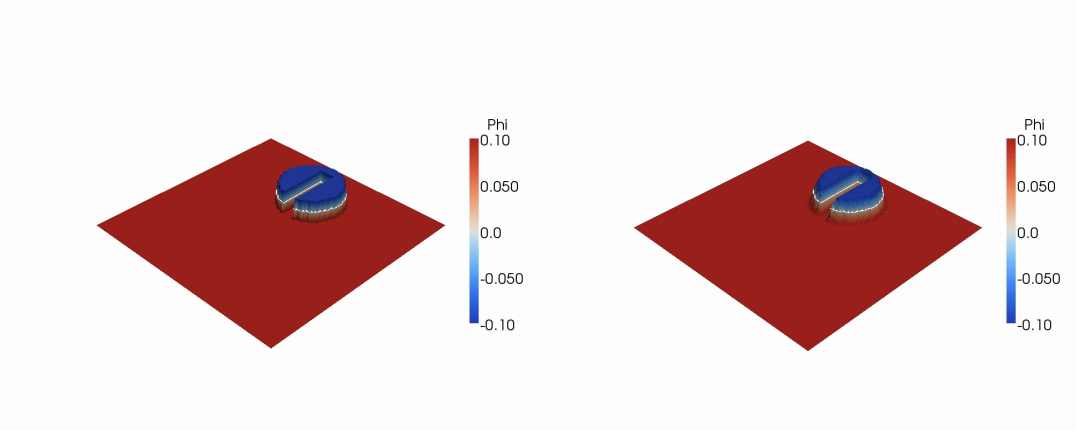
(e) 400 Steps

(f) 500 Steps

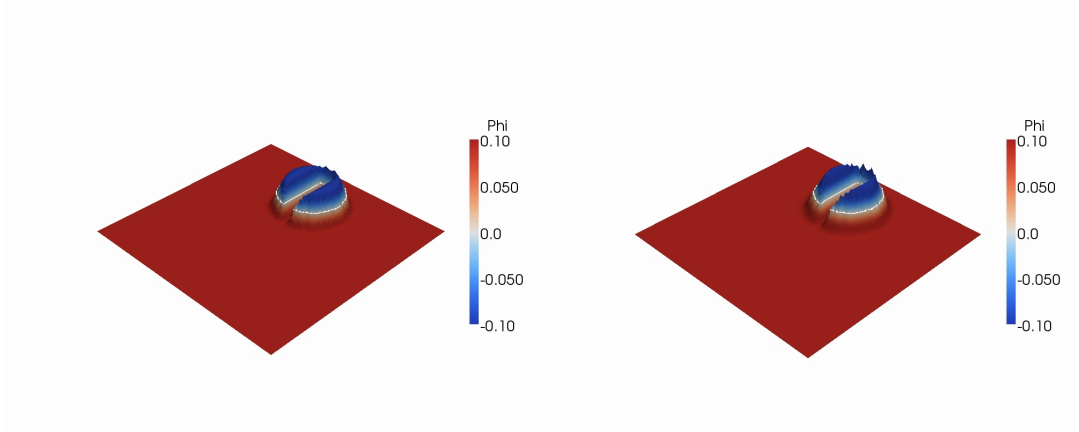
Figure 5.3: Top Hat function. Re-distanced for 500 steps, 101×101 grid resolution

product will always be zero. Upwind gradient approximations do not behave in the same manner. The sample points used for evaluating the gradient are biased to a side of the point being interrogated. Thus, the calculation will return a non-zero gradient estimated from the side of a turning point.

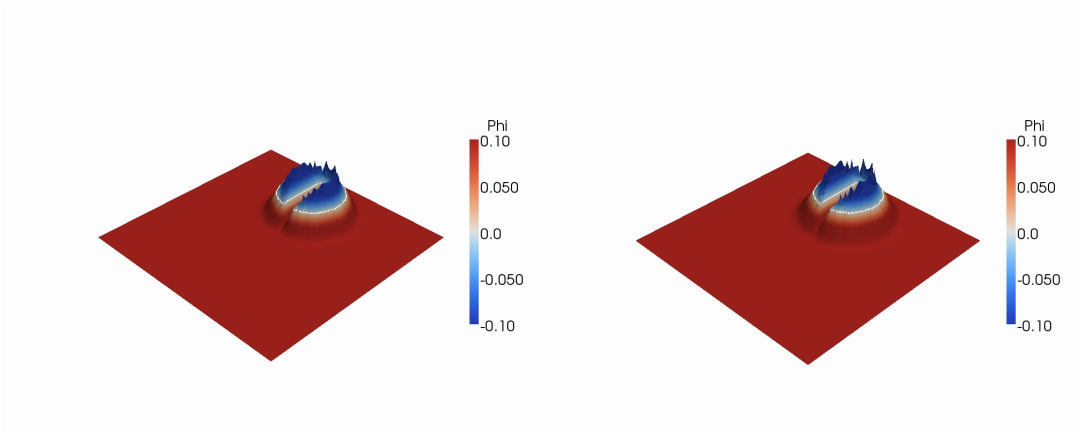
While the static re-distancing tests of the parabolic surface and top hat surface demonstrate reinitialisation spikes occurring on a grid with an odd number of points and symmetrical about a centre point, the third test, of Zalesak's Disc, demonstrates the phenomenon occurring in Figure 5.4. The surface in this figure has been inverted to improve clarity. The level set function is not symmetrical about a single point, instead localised spiking occurs about various points on the surface. This demonstrates that reinitialisation spikes can occur in other, less predictable positions and therefore an adaptable, and a generalised solution is desirable. Such a solution is presented in the following section.



(a) 0 Steps **(b)** 20 Steps



(c) 40 Steps **(d)** 60 Steps



(e) 80 Steps **(f)** 100 Steps

Figure 5.4: Zalesak's disk. Re-distanced for 100 steps, 100×100 grid resolution

5.5 Reinitialisation Spikes and Their Solution

Before proceeding with further test results, and ultimately the conclusion of this Thesis, a short discussion and solution to the reinitialisation spike problem is presented. The section begins with an important remark, entirely relevant to this section, which was made in Osher and Fedkiw (2002, p19),

> *"Equations that are true in a general sense can be used in numerical approximations as long as they fail in a graceful way that does not cause an overall deterioration of the numerical method. This is a general and powerful guideline for any numerical approach. So while the user should be cautiously knowledgeable of the possible failure of equations that are only generally true, one need not worry too musch if the equation fails in a graceful (harmless) manner. More important, if the failure of an equation that is true in a general sense causes overall degradation of the numerical method, then many times a special-case approach can fix the problem."*

While it has been demonstrated that reinitialisation can occur successfully in the previous section, the tests using the new LSL gradient approximation method produced some erroneous spikes in the level set function for certain cases. In the test results the spikes appear to have little influence on the position of the level set close to the position of the zero contour and this reflects personal real world experience during testing. Indeed the spike phenomenon has been mentioned in other similar scenarios by other authors. In Peng *et al.* (1999, p422) the same issue and the difficulty of implementing a generalised resolution is remarked upon. While the test results in Figure 5.2 and Figure 5.3 can have the spike removed by explicitly enforcing a boundary condition at the centre of the domain, Figure 5.4 demonstrates a situation where the spikes form in unpredictable locations. Addressing this case directly is difficult to manage because of the unpredictability of the position of the spikes. Toward this end, Peng *et al.* (1999) offers no solution, relying on the limits of their narrow band level set methods to stop reinitialisation away from the neighbouring band of the zero contour, i.e. in areas where spikes are likely to form.

With advection, effects on the level set surface are greatly reduced. An example of this is displayed in Figure 5.5, a level set function that has been advected for one rotation of the domain. This scenario is significantly different to that of a static re-distancing test. The level set is only re-distanced for one pseudo-time step between each simulation time-step. The problem of spikes forming in this situation is almost totally eradicated as the function is moved across grid each time-step. Nevertheless, a solution to the reinitialisation spike is suggested as it is relatively straightforward to implement should the user wish to, although the details are limited to use on regular Cartesian grids.

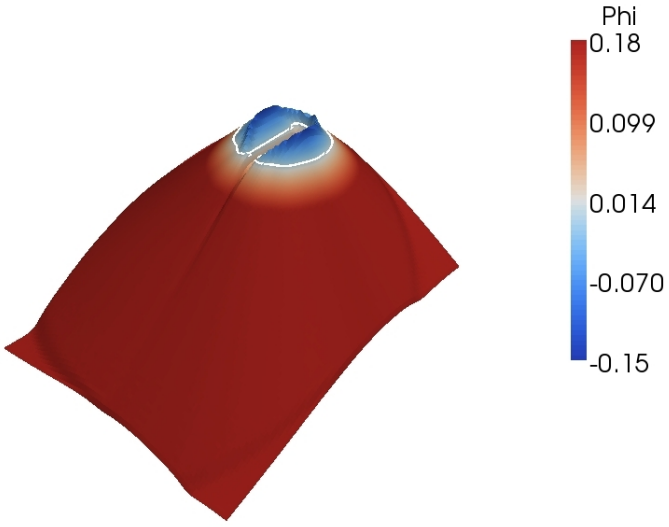


Figure 5.5: Reduced reinitialisation spikes under advection

The initial stage is designed as a simple test to locate the occurrence of a turning point. A 1st order upwind and downwind gradient is calculated,

$$\mathrm{D}\phi_x^+ = \frac{\phi_{i+1} - \phi_i}{\Delta x}, \qquad \mathrm{D}\phi_x^- = \frac{\phi_i - \phi_{i-1}}{\Delta x}. \tag{5.8}$$

The product of gradient approximation signs is then taken,

$$\omega_x = \mathrm{sign}\left(\mathrm{D}\phi_x^+\right) \times \mathrm{sign}\left(\mathrm{D}\phi_x^-\right). \tag{5.9}$$

The test, Equation 5.9, can produce a positive or negative result. In the case of two positive or negative gradients the product will be positive. Only in a case where a turning point occurs and one approximation is positive, the other negative will ω return as a negative value. When a turning point has been found the appropriate 1st order upwind approximation may be substituted for the least squares gradient.

There are two scenarios that constitute a situation in which the 1st order approximation is substituted. The first, generating less conditions for substitution, is to substitute 1st order approximations at points where a turning point in both spatial dimensions is detected. This is more suited to only limiting and removing reinitialisation spikes. This condition is only met and applied in cases where reinitialisation spikes are highly likely to form. The algorithm is as follows, if $\omega_x < 0$ and $\omega_y < 0$ then,

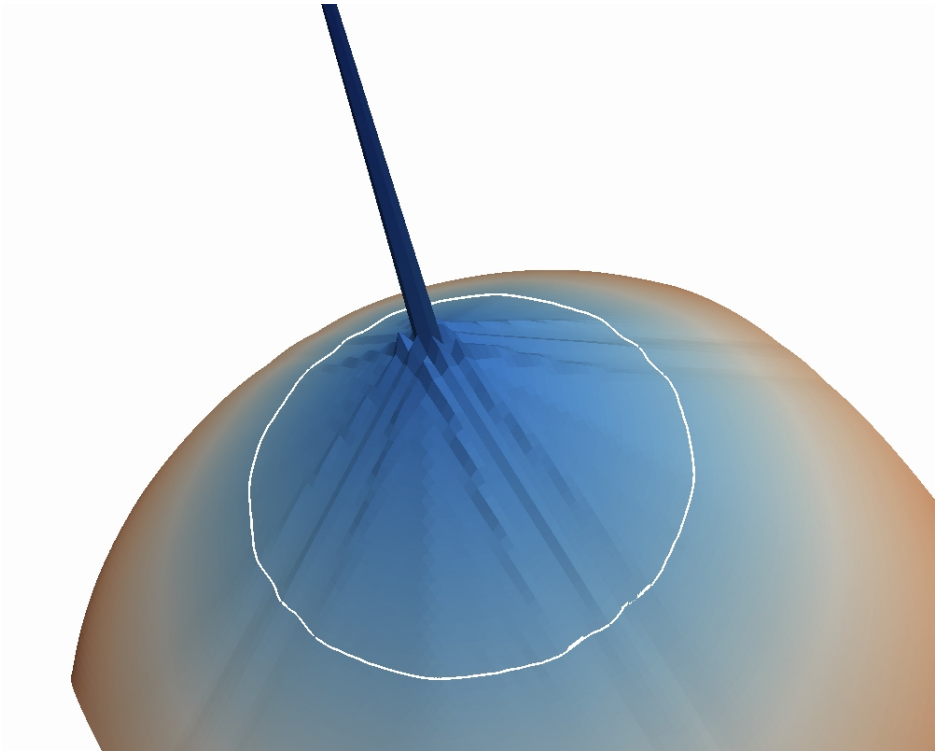


Figure 5.6: Close view of a top-hat level set function re-distanced using no spike suppression, 101 x 101 grid, 500 steps

$$\mathrm{D}\phi_x^{East} = \mathrm{D}\phi_x^{+} \qquad \text{and} \qquad \mathrm{D}\phi_x^{West} = \mathrm{D}\phi_x^{-}$$

$$\mathrm{D}\phi_y^{North} = \mathrm{D}\phi_y^{+} \qquad \text{and} \qquad \mathrm{D}\phi_y^{South} = \mathrm{D}\phi_y^{-}. \tag{5.10}$$

The second approach, the somewhat overly prescriptive method, is to test and substitute for each dimension independently. This approach causes the gradient to be substituted at multiple points on the grid, whenever there is a sign change in a given dimension. While the subsequent results show that in some respects this can be beneficial helping to smooth the surface, it also replaces gradients at points where no initialisation spike was likely to form. It is applied under the following conditions, if $\omega_x < 0$ then,

$$\mathrm{D}\phi_x^{East} = \mathrm{D}\phi_x^{+} \qquad \text{and} \qquad \mathrm{D}\phi_x^{West} = \mathrm{D}\phi_x^{-}. \tag{5.11}$$

and if $\omega_y < 0$ then,

$$\mathrm{D}\phi_y^{North} = \mathrm{D}\phi_y^{+} \qquad \text{and} \qquad \mathrm{D}\phi_y^{South} = \mathrm{D}\phi_y^{-}. \tag{5.12}$$

Examining the most severe example of a reinitialisation spike, the case of statically re-distancing the top-hat function in Figure 5.3 is revisited. The difference between the minimum value and

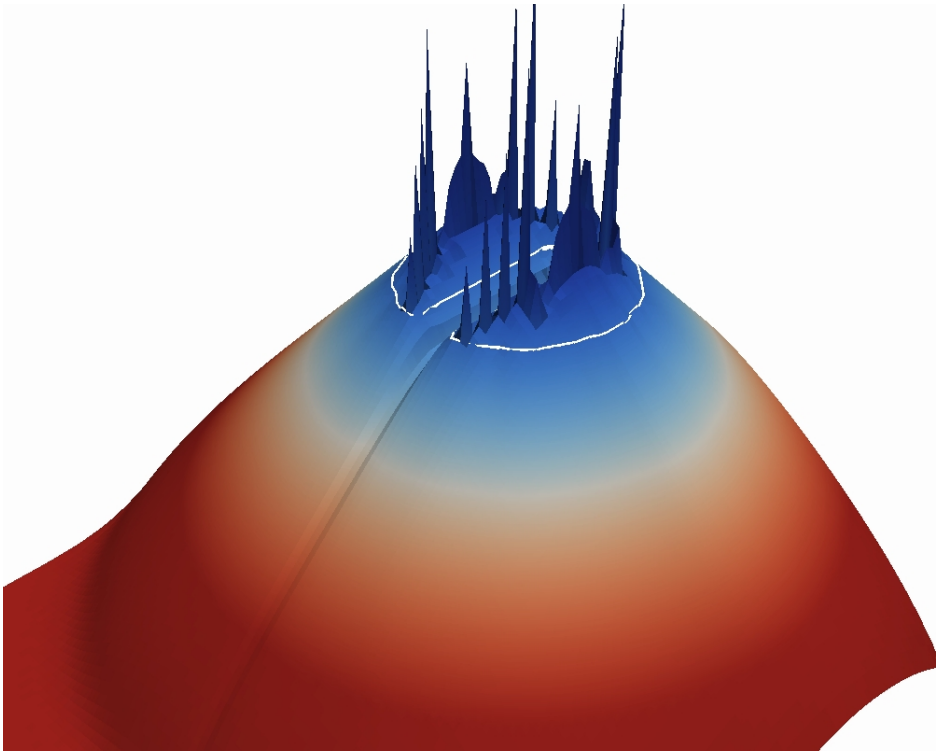


Figure 5.7: Close view of Zalesak's disc level set function re-distanced using no spike suppression, 100 x 100 grid, 500 steps

the zero contour starts large and increases as the function is re-distanced. To re-distance the entire domain, a large number of reinitialisation iterations are required and after 500 of those a severe spike forms.

Figure 5.6 and Figure 5.7 are two inverted, close-up examples of reinitialisation spikes occurring. Both cases have been reinitialised for 500 steps in a static position in order to demonstrate some of the worst case scenarios with which to compare solutions to. The spikes in both figures are severe enough to disrupted the smoothness of the level set function. Figure 5.6 has some loss of smoothness around the centralised spike and corrugated surface features leading to the surface in the separate axis directions. None of these features however, significantly disrupt the position of the zero level set contour (marked in white) although it is apparent that they are on the limit of doing so.

Figure 5.7 is another example of the reinitialisation spike phenomenon in the extreme. This example is intentionally over reinitialised, in the sense that it only should have required 100 steps to adequately re-distance the surface to a signed distance function. The re-distancing has, however, been allowed to continue for 500 steps causing the severe reinitialisation spikes to occur. These spikes are so bad that the contour of the zero level set is starting to be disrupted in position.

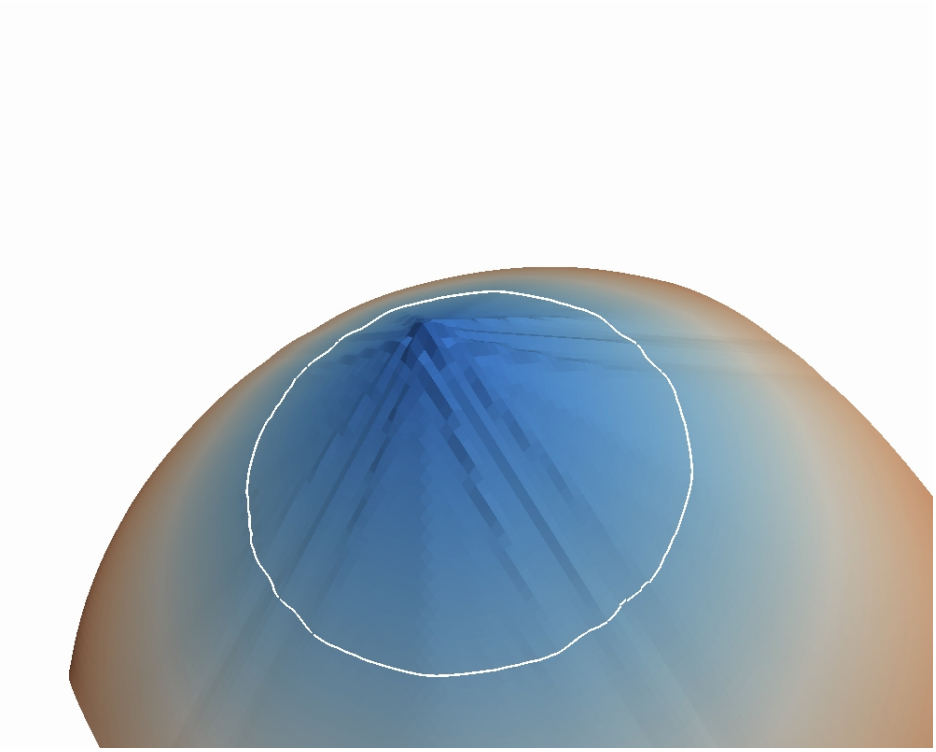


Figure 5.8: Close view of a 'top hat' level set function re-distanced using light spike suppression, 101 x 101 grid, 500 steps

Figure 5.8 and Figure 5.9 are the examples demonstrated with the first method spike suppression. That is, only when both x and y gradients indicate a turning point are the gradients substituted for 1^{st} order upwind approximations. The figures clearly show that this approach works effectively. All instances of spikes are suppressed sufficiently and while there is still a lack of smoothness to Figure 5.8, the spike at the centre of the domain has been removed. In Figure 5.9, the severe spikes have also been removed, while the crest on the inside of the disc is not entirely smooth, the gradients are much closer to the desired value of 1 and the spikes no longer risk adversely affecting the shape of the zero contour.

Finally applying the second criterion to the gradient substitutions, Figures 5.10 and 5.11 use the more severe method of spike suppression. In this case the example of the re-distancing of the top-hat function shows that not only has a spike been suppressed from the centre of the domain but that the corrugated surface of the level set has been smoothed. In a similar manner the 'crest' of the level set function on the inside of the Zalesak's disc example is smoothed further. The smoothing effect seen using this approach is due to the numerically diffusive nature of the 1^{st} order approximation being substituted in. While this is strictly not the most accurately re-distancing method, the user may wish to have the benefit of the smoother level set surface.

The above discussion could be considered a moot point. When referring to Figure 5.5, the Zalesak's disc has been advected and reinitialised for one rotation of the domain and is of

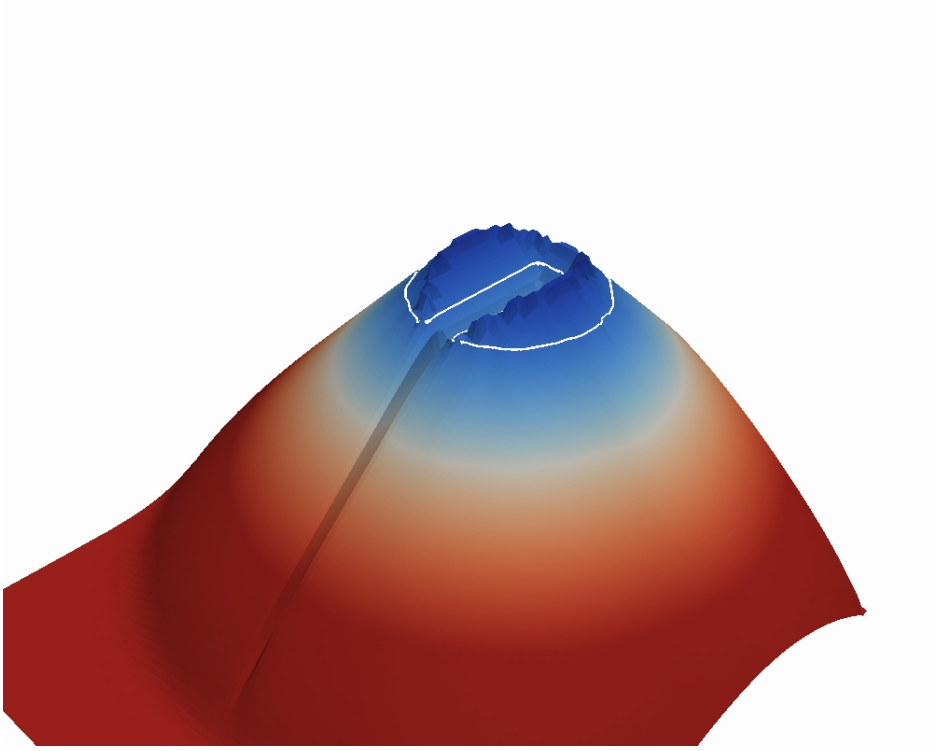


Figure 5.9: Close view of Zalesak's disc level set function re-distanced using light spike suppression, 100 x 100 grid, 500 steps

comparable smoothness and quality to one of the spike limited static reinitialisation examples. The act of simply moving the level set function on the numerical grid, ensuring the same turning points are not repeatedly reintialised is enough to prevent the problem occurring for the vast majority of normal use.

This concludes this brief aside into the suppression of reinitialisation spikes, whilst using centrally differenced based gradient approximations. Should the user wish to, one of the approaches can be included in the level set reinitialisation scheme. However, as the results in the remainder of this PhD Thesis only require a few reinitialisation iterations applied between each time-step, no spike suppression is implemented for the results found in Chapter 6.

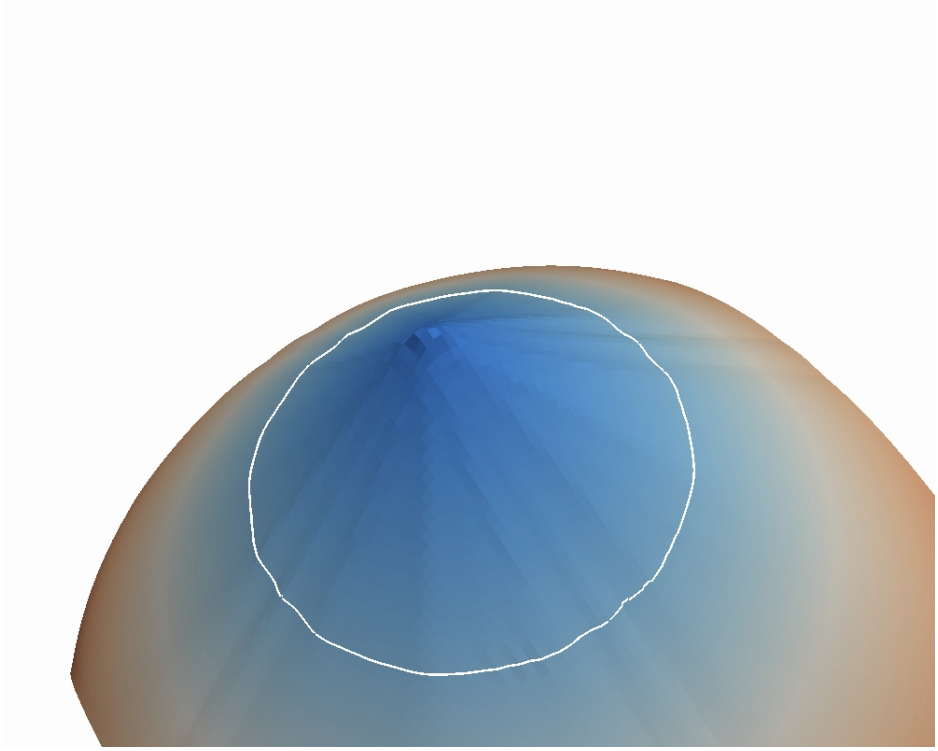


Figure 5.10: Close view of a 'top hat' level set function re-distanced using heavy spike suppression, 101 x 101 grid, 500 steps

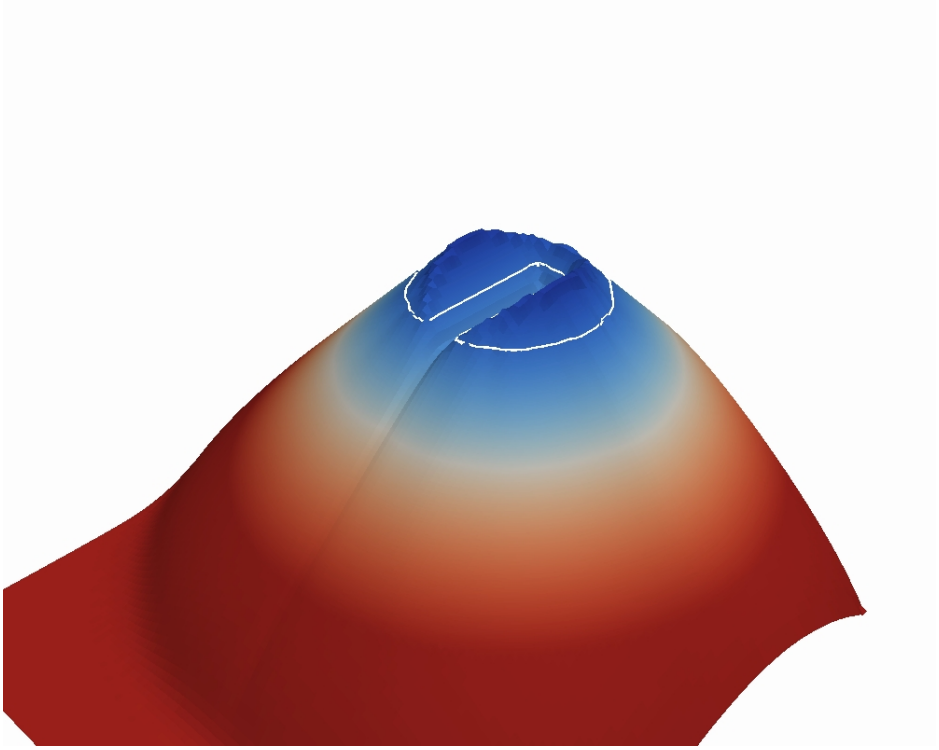


Figure 5.11: Close view of Zalesak's disc level set function re-distanced using heavy spike suppression, 100 x 100 grid, 500 steps

5.6 Extending the Scheme to an Unstructured Grid

This section summarises a naïve attempt at extending the slope limiter to operate on an arbitrarily structured grid. A brief discussion on the assumptions behind this approach and why it would be successful is followed by the required changes to the gradient limiting method and the results it generates. Based on the conclusion of this section, a description of how the slope limiter could be successfully implemented on an unstructured grid is included in Section 5.6.2.

The goal of this PhD project was to develop a reinitialisation scheme to provide better than 1^{st} order accuracy for unstructured grids of generalised geometry, such as Cartesian cut-cell. Choosing to use the highly adaptable least squares method requires the implementation of a gradient limiter to prevent gradient overestimate during reinitialisation and subsequent loss of solution stability.

The challenge faced by extending the scheme to unstructured grids is that there is no longer a succession of points aligned with the domain axis, a constraint also discussed in Jasak *et al.* (1999). Neighbouring points occur in arbitrary locations about each point. For a 2D scenario, this makes it impossible to limit gradients based on the upwind 1^{st} order approximation using components in x and y. Referring to Figure 5.1, the only information available for calculation is a gradient magnitude from point P to point N and a position vector from a point to its neighbour. This makes the procedure difficult as the method detailed in Equation 5.3 is no longer valid. The following solution was thought to be one method to simplify the limiting process, allowing it to be used with any orientation of neighbouring points. After investigation, it became clear that the behaviour of the scheme was misunderstood and an improved method is described in Section 5.6.2.

5.6.1 A Slope Limiting Method Using Approximate Local Data

As it is not possible to calculate the x and y components of gradient to complete the Godunov Hamiltonian, and calculate the gradient magnitude, the optimal solution would be to use a point that intersected the level set normal at some location and use it to limit the gradient magnitude produced by the least squares approximation. The first part of the solution requires the area's space divided into quadrants, displayed in Figure 5.12. It was assumed that the stability of the reinitialisation could be maintained if the neighbour points in an upwind quadrant were found, and the gradient magnitude of the level set function was limited such that a projection to the selected neighbour points did not exceed their original value. This assumption was based on the fact that the under estimate of gradient during the reinitialisation iteration simply results in the level set function not being adjusted by a sufficient value. While this may result in loss of accuracy locally, it does not create the numerical instability that an overestimate of gradient is capable of.

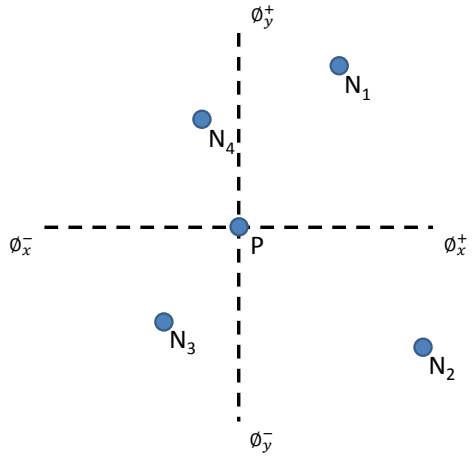


Figure 5.12: Splitting the spatial domain into four directional quadrants

In a 2D domain there are four quadrants, defined by the upwind and downwind directions of the x and y axes. A selection procedure can be used to select the correct quadrant for use when limiting the result of the least squares scheme, using a variation of the selection criterion suggested in the paper by Sussman *et al.* (1999). As there are four quadrants to select from and quadrant selection is affected by the sign of the level set function, this leads to eight cases that may be selected. This increases to 16 in 3D. To avoid using inefficient logic-heavy computing routine a simple sign test is used to reduce complexity of the logic procedure.

In each dimension, with x shown as an example,

$$\gamma_x = \text{sign}\nabla\phi_x \cdot \text{sign}\phi. \tag{5.13}$$

Now that the signed product γ has been generated in each dimension, select a case from 4 that exist,

$$\text{Quadrant} = \begin{cases} \phi_x^+\phi_y^+ & \text{if} \quad \gamma_x > 0 \quad \text{and} \quad \gamma_y > 0 \\ \phi_x^+\phi_y^- & \text{if} \quad \gamma_x > 0 \quad \text{and} \quad \gamma_y < 0 \\ \phi_x^-\phi_y^+ & \text{if} \quad \gamma_x < 0 \quad \text{and} \quad \gamma_y > 0 \\ \phi_x^-\phi_y^- & \text{if} \quad \gamma_x < 0 \quad \text{and} \quad \gamma_y < 0 \end{cases} \tag{5.14}$$

this is the quadrant that the level set unit normal projects into and contains the point that should be used for basing the limiter coefficient on.

In the case of the Cartesian grid that testing has been conducted on, two points lie ambiguously on the border between quadrants. Therefore, to apply this new slope limiting method to a regular Cartesian grid represents a worst case scenario for limiting the level set function gradient.

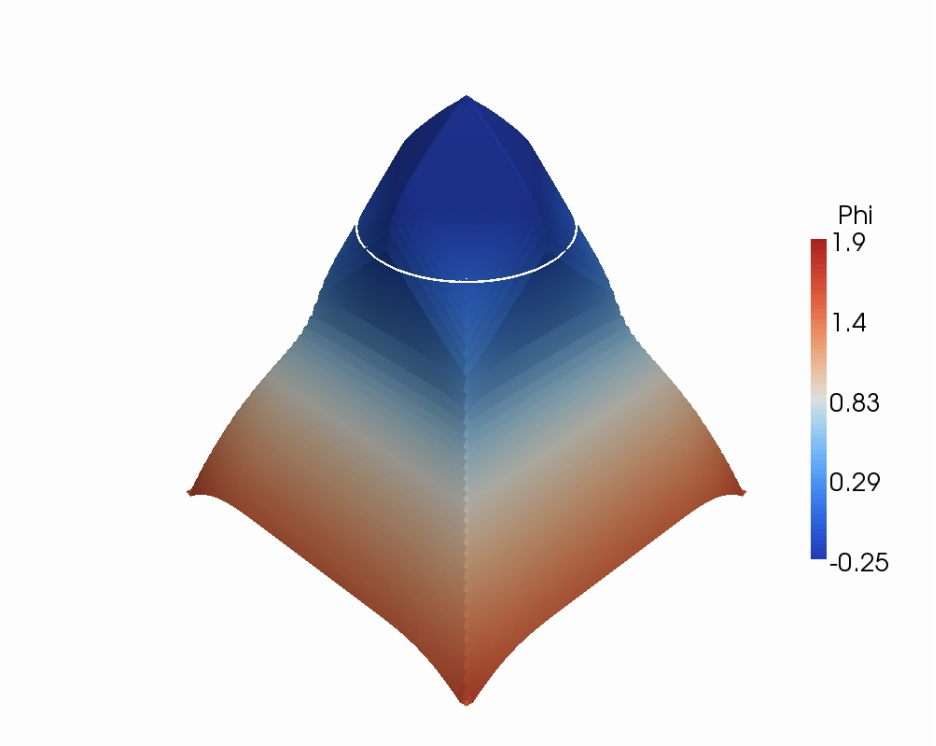


Figure 5.13: Parabolic surface test. Test gradient limiter routine (broken). 200 re-distancing steps

The two points are likely to be placed some distance from the a position described by the normal of the level set function. The gradient for the level set reinitialisation is limited to these points. The final reinitialised surfaces are displayed in Figure 5.13.

As can be seen, this approach has been demonstrated to give an undesirable result. The level set function is manipulated to achieve a surface that has a gradient of magnitude 1 with a normal aligned with either x or y axis. The surface loses any gradient in the other axis direction and the zero contour becomes distorted. By limiting values to the minimum of the two points falling in each quadrant, the level set function it is effectively limited to either the gradient in x or y giving the surface its square sided appearance. It is possible to see the change of selected limiting point, as the sign of ϕ changes resulting in the rotation of the edge through 90^o. Clearly, it can be deduced from this that future attempts to limit the gradient magnitude must be based on comparing gradient values that lie in the direction of the normal. One such method is discussed here.

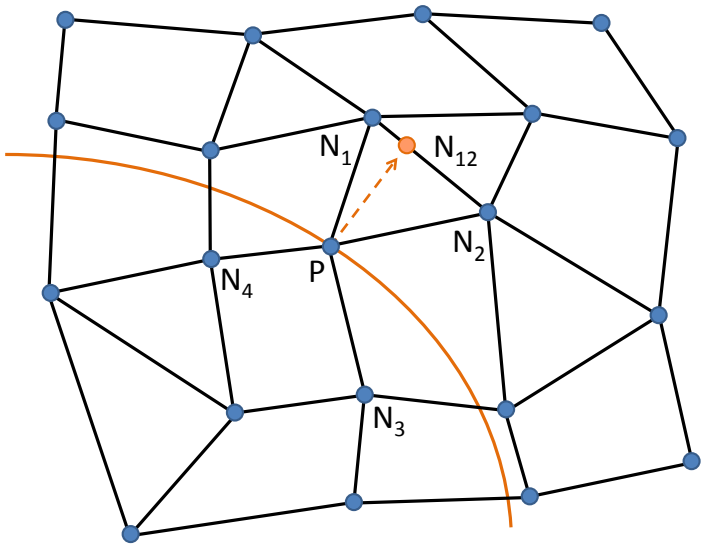


Figure 5.14: Interpolated point to grid

5.6.2 An Improved Slope Limiting Method for Unstructured Grids

It has been established that any limiter applied to the gradient of the level set function must use data from a point that intersects with the function normal (shown in Figure 5.14). As the individual partial derivatives of the function cannot be easily calculated on an unstructured grid an alternative approach must be found. The method suggested has not been tested on an unstructured grid given the lack of an easily integrable framework with the current work. It is reasonable to believe that with further research effort some of the ideas and methodology here could be incorporated into a flexible, robust method for use on Cartesian cut-cell grids.

It is straightforward to establish the normal vector of the level set function using $\mathbf{n} = \nabla\phi/|\nabla\phi|$. By using the equation 5.14 to establish whether the the point of intersection should be calculated upwind or downwind of the gradient normal it is possible to establish the position of a data point that can be used for gradient limiting.

To do so will allow the calculation of the intersection point N_{12}, depicted in Figure 5.14. That point intersects the level set normal and its value of ϕ may be found through interpolation on the cell face F_{12}. To guarantee the monotonicity of the reinitialisation method, this interpolation must be conducted before reinitialisation is completed, effectively reconstructing the level set function at new grid points that lie directly on the level set normals. Therefore the slope limiter equation can be rewritten as

$$\psi = \frac{\phi_{N_{1,2}} - \phi_P}{|\nabla\phi_P \cdot \mathbf{r}|} \tag{5.15}$$

where $\mathbf{r}$ is the position vector to the point N_{12}.

The gradient in the direction of the normal can be successfully limited using

$$D\phi = \nabla\phi \cdot \mathbf{min}(\psi_{N_{i,j}}, 1). \tag{5.16}$$

An interpolation method similar to the suggestion of Darwish and Moukalled (2003) is a potential solution to the reconstruction of the grid values. While interpolation is a potential source of error, if this method was considered principally for implementation on a Cartesian cut cell grid, there would be little issue with a loss of accuracy in the small percentage of unstructured cells.

5.7 Concluding Remarks on the LSL Reinitialisation Scheme

For the first time, to the author's knowledge, a novel application of the least squares approximation scheme has been presented in this chapter. The use of a new slope limiting procedure has been developed to guarantee monotonicity in a manner similar to other TVD schemes, making the scheme suitably stable for use in a level set reinitialisation scheme. This method has the potential to work with adaptive grids, as others (Wang and Wang, 2004) have successfully used the least squares method on such grids. However the limiting method presented here has the additional potential to be extended to grid with irregular geometries.

An undesirable side effect of the scheme, is a unpredictable creation of reinitialisation spikes during repeated iterations of the reinitialisation scheme at a single time level. The cause was found to be due to the indeterminate gradient at turning points when calculating an approximation using a centrally biased scheme, such as least squares. A new method, with several variations, was developed to identify potential problem points in the numerical grid and remove the potential for reinitialisation spikes to develop. However, it was found that in general advection problems, the reinitialisation spike issue was not prevalent due to the low number of reinitialisation iterations used at each time-step.

Finally a scheme that used a value for the limiter obtained from a local data has been trialled on a structured grid. The approach was found to be unsuccessful, with distortion of the level set signed distance. Notably, the position of the interface remained fairly accurate even in this scenario. An amended method involving reconstructing the data of the level set at points lying in the path of the level set normal was suggested. Using an interpolation approach to the method put forward by Darwish and Moukalled (2003), it should be possible to obtain a successful result on an irregular geometry grid.

Chapter 6

Comparison of Reinitialisation Methods

In Chapter 3 a CIP advection solver was presented and demonstrated advecting a level set function in a 2D cylinder test. In Chapters 4 and 5 a variety of methods for level set reinitialisation and maximising its accuracy have been presented. To conclude the main body of work in this Thesis a comparison of level set methods is conducted. The 2D cylinder advection test case will be assessed in five different scenarios. These are advection with no reinitialisation; both advection with 1^{st} order reinitialisation and reinitialisation using the improved, area conserving algorithm; advection with the new reinitialisation method using the least squares limited scheme and again that scheme with the improved area conserving constraint. The goal is to establish the best performing scheme of the five.

In the second section of this chapter, the leading reinitialisation scheme is assessed to find its optimum performance in the 2D cylinder test, therefore finding the most computationally efficient and accurate method of implementing the best performing scheme. Having found appropriate parameters for reinitialisation, a comparison of best performance scenarios and computation times is investigated. These scenarios give the reader some idea of the benefits to accuracy a reinitialisation scheme can bring to the level set method.

The final inclusion in this chapter is a more challenging test case, Zalesak's Disc. The disc is a circular contour with a sharp cornered slot cut into it. Through numerical diffusion during advection, poorly performing methods reduce the sharpness and depth of the slot in the disc. The test case is qualitatively assessed for how effectively level set reinitialisation can improve the accuracy of its shape. Conclusions made from the test are expanded on in the future work section of Chapter 7. The performance demonstrated in this chapter demonstrates the potential for this scheme to warrant its further development towards unstructured grids.

6.1 Testing Methodology

In Chapter 3 a grid convergence study based on the amount of area preserved inside a circular contour represented by a level set function was introduced. Results showed that converged results were achieved on a grid resolution of 385×385 or higher. In the following comparisons there will be two measures of the quality of the interface preservation recorded. Area or mass preservation is the first, which has been used for the convergence study and is important for engineering problems. The second, and of equal importance, is the accuracy of the shape of the contour.

To quantify the contour shape, a new measurement is introduced for this chapter and is referred to as the 'contour skew'. The premise of the measurement is based on the fact that a circle encloses a greater area than any other shape of a fixed perimeter. Therefore if the contour perimeters are compared before and after advection, an increase in perimeter indicates a skewing of the contour from perfect circularity to an elliptical shape.

The technique is slightly more complex, requiring one final step due to the inevitable loss of area during advection. Area loss has the natural consequence of a reduction in the perimeter of the contour and removes the possibility of a before and after comparison. It is therefore necessary to calculate the final area of the contour before proceeding to calculate the perimeter of a perfect circle of that area. It is this calculated perfect perimeter that can be compared to the measured perimeter at the end of advection. A contour skewing calculation based on percentage perimeter elongation can be made and used for comparison between the various numerical schemes. The procedure is summarised in Equation 6.1.

$$c_{perfect} = 2\pi\sqrt{\frac{\text{Area}}{\pi}},$$

$$\text{Contour skew} = \left(\frac{c_{measured}}{c_{perfect}} - 1\right) \times 100\%, \tag{6.1}$$

where $c_{measured}$ and $c_{perfect}$ are the measured and perfect circumferences of the contour enclosing the final area respectively.

This method provides a reliable and measurable difference for measuring the shape of the final contours. It must be noted that this contour skew measurement is extremely sensitive; as little as 1% skew indicates a contour that is visibly elliptical.

As time integration schemes are not the primary focus of this project, and all methods have used a 1^{st} order accurate scheme, little work is spent examining this aspect on the accuracy of the results. All tests are conducted at the settings to allow for fair comparison using a CFL number of 0.5. In the first area of investigation, performance of the schemes on grids of resolution 49×49 to 769×769 are compared for both area conservation and contour skew. The goal of

this section is to clearly demonstrate than the new LSL approximation schemes perform to a higher accuracy the other schemes presented.

The second stage of investigation examines the results at two resolutions, specifically a solution that was shown to be under resolved in Chapter 3 on a grid of 193×193 points and at a fully converged 385×385 resolution grid.

The third set of tests results in this chapter are a short study on sensitivity of the best performing reinitialsiation scheme to varied parameters. With this approach, the most efficient, best performing settings for the reinitialisation may be found. The optimised results are used to demonstrate that for a given setting more accurate answers can be obtained using level set tracking with reinitialisation and the new LSL gradient scheme than with level set tracking on a higher resolution grid.

In the final test of the chapter, the classic test known as Zalesak's disc is performed. A qualitative approach is taken to assessing the performance of this test, comparing unreinitialised and reinitialised versions of the test.

6.2 Comparison of Advection Accuracy Using Various Reinitialisation Schemes

In this section, the advection problem of the rotating 2D cylinder, introduced in Chapter 3, is analysed for area conservation and shape preservation using five cases: advection with unreinitialised advection 1st order reinitialisation, 1st order improved reinitialisation, 2nd order reinitialisation and 2nd order improved reinitialisation. A full grid convergence study is omitted in this chapter as the same underlying advection scheme, tested in Chapter 3 is used throughout all five schemes. It is therefore no surprise that all four reinitialised schemes display the same order of convergence as the unreinitialised advection scheme. The comparison of methods uses parameters originally suggested in Sussman *et al.* (1994)[p154], that is one reinitialisation iteration per time-step of the advection solver using a pseudo time-step of $\Delta t = h/10$.

Figure 6.1, a plot of Area loss vs. Grid resolution, compares the five advection cases for their area conserving ability. Of all the reinitialisation methods, only the new least squares limited method performs better than the original unreinitialised advection problem. The improved reinitialisation scheme using LSL approximation performs well, being extremely close to the unreinitialised scheme in area preservation. The remaining two 1st order reinitialisation schemes show undesirable performance with the improved reinitialisation scheme, underperforming significantly.

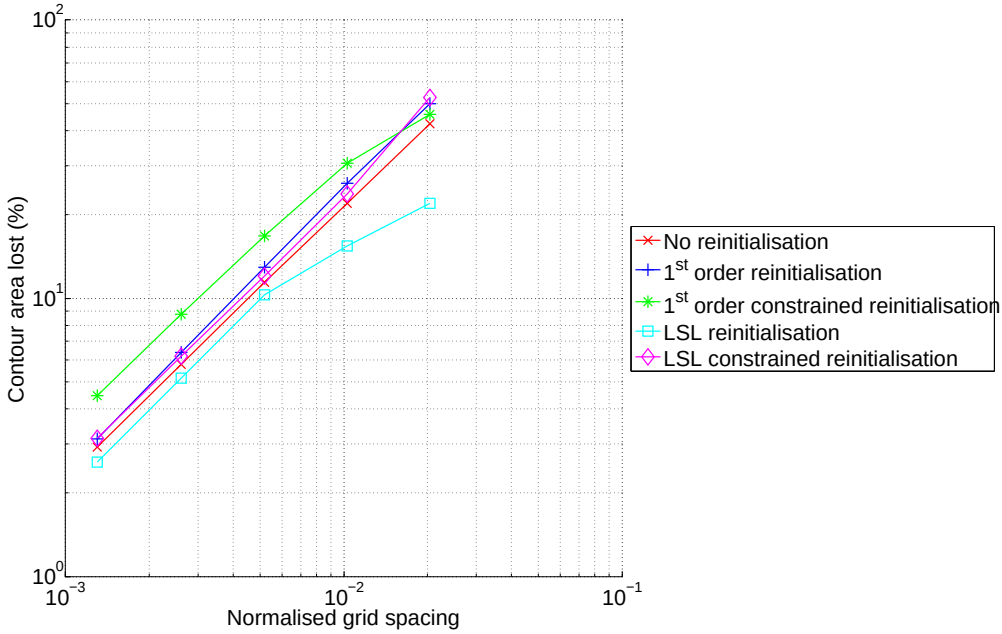


Figure 6.1: Area loss vs. grid resolution for five advection cases

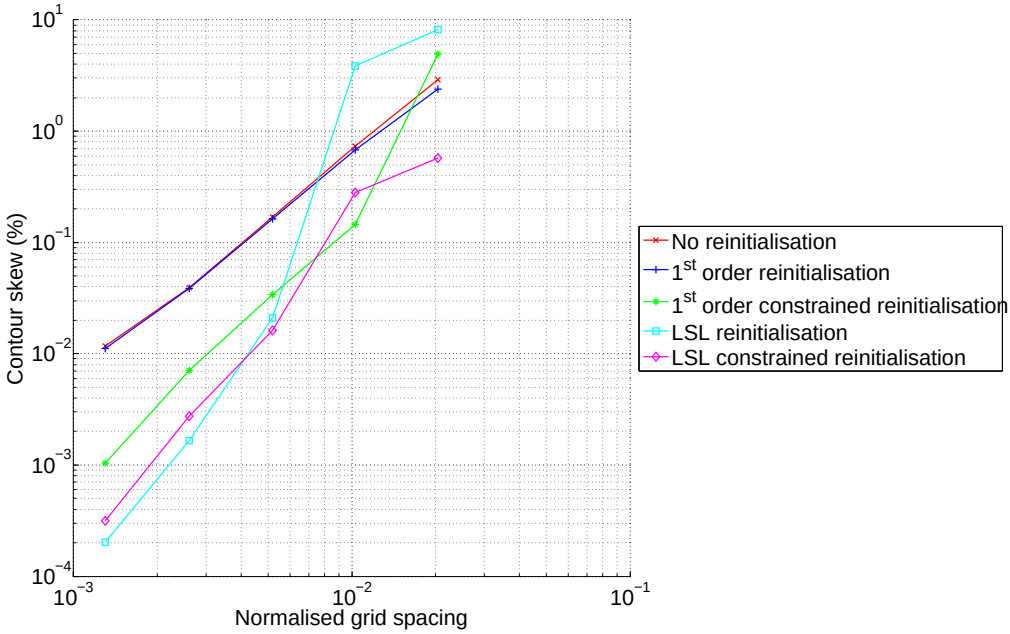


Figure 6.2: Contour skew vs. grid resolution for five advection cases

The most likely cause of such poor area preservation for a reinitialisation scheme that was specifically designed to conserve area is due to the low order discretisation affecting the accuracy of the local area conserving constraint. More discussion on this performance follows in the proceeding sections when more results have been presented.

In Figure 6.2 the accuracy of the contours shape is compared using the skew measurement described in Section 6.1. It can be seen that all of the reinitialisation schemes perform as well or better than the unreinitialised CIP advection scheme, provided results are taken from an adequately resolved grid.

Discarding the 1st order reinitialisation scheme, all three remaining schemes have a positive effect on the accuracy of the shape of the contour. At grid converged resolutions of 385×385 and above, again the 2nd order reinitialisation without constraint performs best and when the constraint is implemented, performance is similar and better at under resolved grid resolutions.

Figures 6.3 and 6.4 present the lost area vs contour skew data as a scatter graph at two different resolutions. This is essentially the same data as Figures 6.1 and 6.2, each at a single resolution to enable a clearer comparison of the individual schemes for the reader.

These graphs were the first to be plotted when analysing the original results of each of the reinitialisation schemes combined with the CIP advection scheme. Both the plots from a spatially converged grid and an under-resolved grid are included. The choice to include the 193 grid

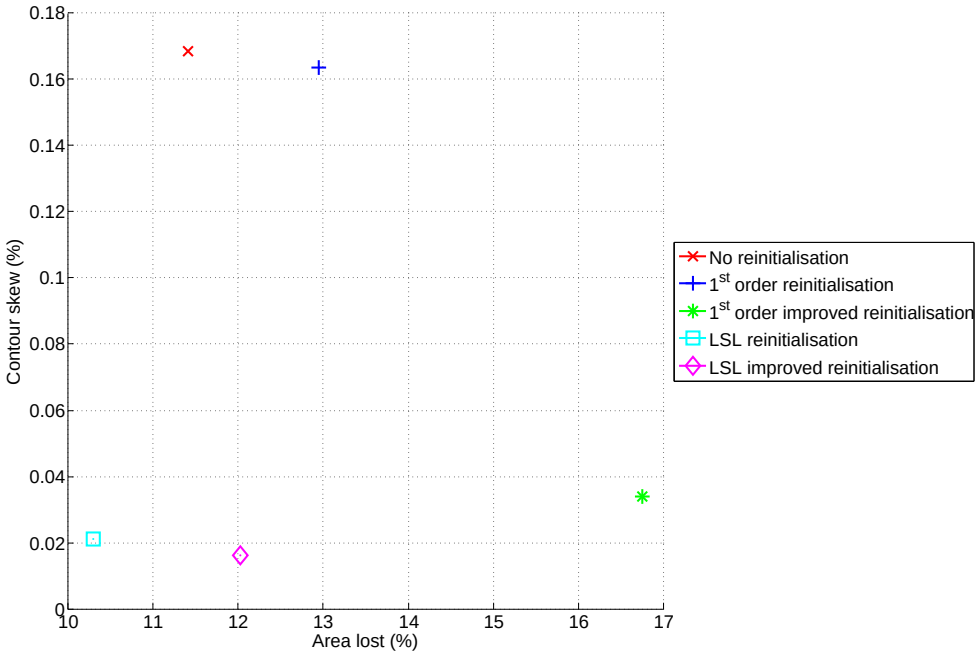


Figure 6.3: Comparison of advection cases on a 193×193 grid

was to determine if the reinitialisation helped to significantly improve the accuracy of the level set method in under-resolved situations. The positioning of each schemes results for the data points at the two resolutions is extremely similar. In each plot it can be seen that the methods using the new LSL gradient approximation outperform their counterparts in shape accuracy with little impact to their ability to conserve area.

The unreinitialised and the simple 1^{st} order scheme's results are closely grouped together demonstrating that the 1^{st} order scheme is not accurate enough to improve the accuracy of the contour shape and it loses more area in comparison to the unreinitialised results. While there is no doubt that the improved 1^{st} order reinitialisation scheme is able to improve the contour skew value this method contributes to area loss of the simulation and cannot be considered useful. Upon reflection, it is likely that some of the improved contour skew performance is down to the diminishing area held by the contour and not improved accuracy.

The new schemes perform significantly better than the unreinitialised advection test in terms of shape accuracy. However, there is a slight tendency for the reinitialisation methods to move the zero contour such that area is lost from inside the 2D cylinder. Although the reinitialisation method using the area preserving constraint showed some signs of improving the accuracy of advection when using 1^{st} order gradient approximation it does not conserve area as successfully when using the LSL scheme.

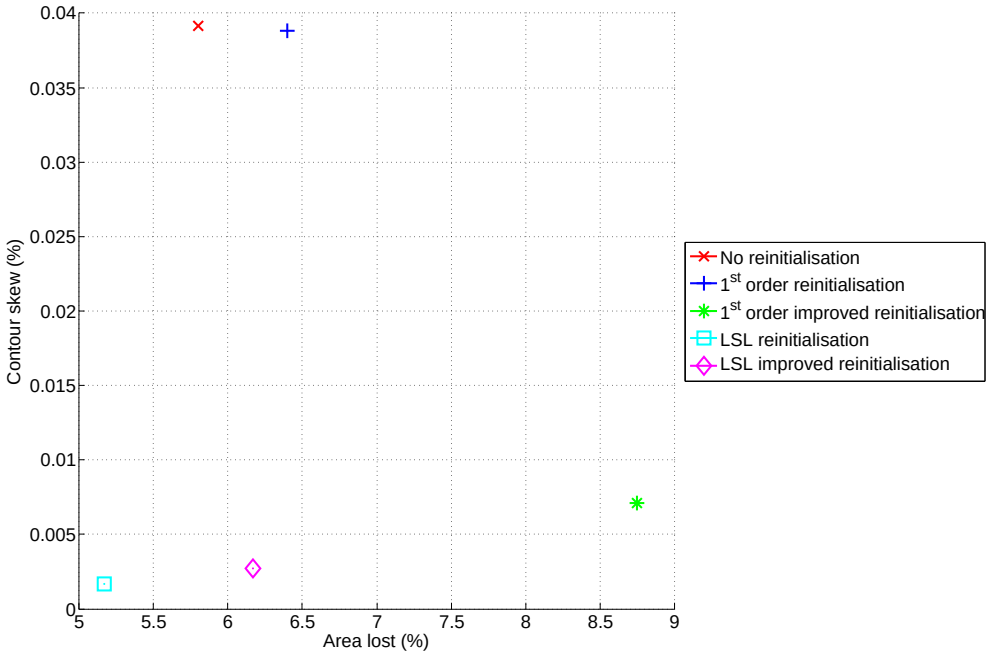


Figure 6.4: Comparison of advection cases on a 385×385 grid

Having established that the new least squares limited approximation scheme presented in this Thesis performs with an increased level of accuracy, some time will be spent in the next section examining what parameters should be chosen for its optimum performance for the test case.

6.3 The Optimisation of the Least Squares Limited Reinitialisation Method

Some further investigation is conducted to compare the effect of varying different parameters of the reinitialisation method. The parameters that will be altered are the three that are directly related to the reinitialisation method; the CFL number of the pseudo time-step, the frequency between simulation time-steps that a reinitialisation iteration is made and the number of reinitialisation iterations made between time-steps. At the end of the study an optimal combination of settings that maximise accuracy and minimise computational time will be found.

The following tests are completed on the 193×193 grid using the best performing reinitialisation scheme, the new LSL reinitialisation scheme without the area preserving constraint. The 193×193 grid was chosen due to the fact there were no significant differences in the behaviour of the reinitialisation scheme at higher resolutions. The opportunity to run the tests for a shorter computational time is a benefit that many users desire. While this study focuses on optimisation for a specific test case, the observations made in this section are equally valid for a more general range of simulation scenarios.

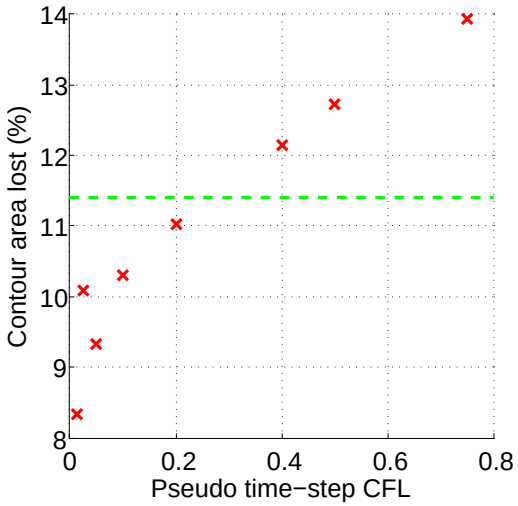
6.3.1 Varied CFL Number for Pseudo Time-step

In the first two figures, Figure 6.5, the effect of varying the pseudo time-step is displayed for area loss (Part (a)) and contour skew respectively (Part b). During these tests level set reinitialisation takes place between every simulation time-step. The green line included in the plots shows the value produced by the unreinitialised level set value obtained on the same resolution grid.

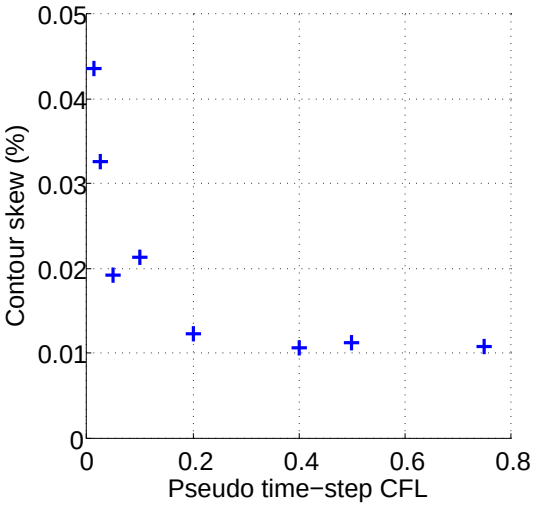
Part (a) of the figure shows that as the pseudo time-step is increased in magnitude the area lost inside the cylinder increases. Above of CFL number of 0.1 the area loss increase appears entirely linear which is expected due to the 1^{st} order time integration scheme used in this reinitialisation algorithm. Below a CFL of 0.1, results show less of an consistent trend but generally show a continual reduction in area loss at lower CFL numbers.

The accuracy of the contour shape is plotted in Part (b) of Figure 6.5. The contour skew and therefore shape accuracy is better than the unreinitialised result in all cases and improves with increasing pseudo time-steps. The green reference line of the unreinitialised result is not included on this plot, being at a much higher value it is far off the scale of the graph.

From these results, it can be surmised that during level set reinitialisation the contour is moved fractionally, resulting in a reduction in area inside the zero contour. The extent of this area loss is directly proportional to the pseudo-CFL number for cases with that number above 0.1. This is not an unrecognised phenomenon as the gradient approximations used during the reinitialisation procedure take information from the wrong side of the zero level set contour during



(a) Pseudo time-step CFL number vs. Area lost



(b) Pseudo time-step CFL number vs. Contour skew

Figure 6.5: Area lost during contour advection vs. Pseudo time-step CFL number

reinitialisation as addressed by Russo and Smereka (2000). It is therefore beneficial to limit the the reinitialisation time-step to a low number such as 0.1 as originally suggested in Sussman *et al.* (1994).

6.3.2 Frequency of Reinitialisation Iterations

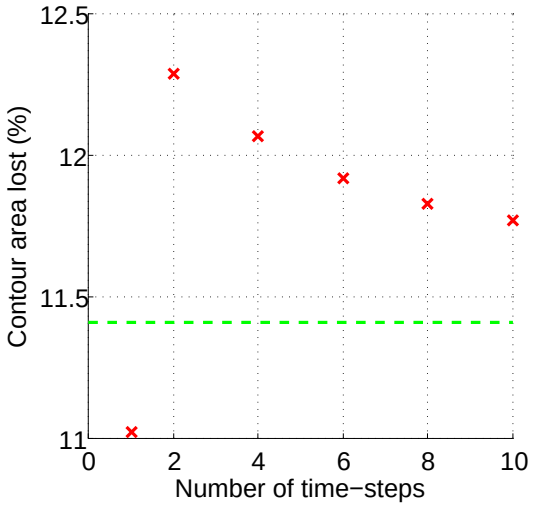
In the next section of the study the effect of varying the number of simulation time-steps between which the reinitialisation routine is called is examined. Figure 6.6 shows the effect of the varying frequency, again Part (a) shows the effect on contour area and Part (b) shows shape accuracy with the green line highlighting the value for an unreinitialised simulation. The reinitialisation method was used with a pseudo time-step CFL of 0.2 throughout, for consistent comparison.

Part (a) of the figure shows that a reinitialisation iteration every time-step preserves contour area more effectively, by 0.7%, than other frequencies of reinitialisation. Decreasing the reinitialisation frequency to once every two time-steps has a strongly detrimental effect on the area preservation. As the number of time-steps are increased, the area loss effect decreases, appearing to converge on a value. It is reasonable to assume this value would the same as the area loss of the unreinitialised test case.

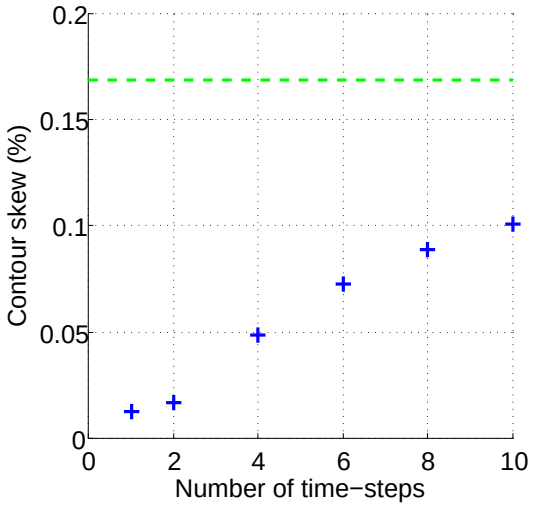
The accuracy of the contour shape decreases the more infrequently the reinitialisation routine is called with reinitialisation between every time-step yielding the best results. Again, it is reasonable to assume that as the frequency of the reinitialisation routine decreases, the contour skew figure will converge on the figure generated by the unreinitialised problem.

In this scenario, executing the reinitialisation routine between every time-step has been shown to produce the best results. While this was already stated to be the case by Sussman *et al.* (1994) it is important to reconfirm this using the new 2^{nd} order gradient approximation method. An interesting feature that deserves more comment is the large jump in area loss moving from one to two time-steps frequency. The evidence suggests that allowing an extra time-step between reinitialisation iterations has a sufficiently detrimental effect on the accuracy of the signed distance to impede the reinitialisation method's ability to redistance the function without moving the level set contour. It is therefore better to either redistance every time-step maintaining extremely high accuracy or not use a reinitialisation routine.

A valuable observation is clearly, reinitialisation every time-step has a large impact on the ability of the scheme to conserve the contour area. Figure 6.5 has shown that reinitialisation has a tendency to move the level set interface by a small amount. Allowing the level set function to lose its signed distance property, by even a small amount, after two or more time-steps causes reinitialisation to be less accurate and consequently move the level set interface more, reducing the cylinder area. Obviously the less reinitialisation procedures that are executed in the simulation the less area is lost, reflected by the graph. What is surprising is this effect is



(a) Area lost vs. Number of time-steps



(b) Contour skew vs. Number of time-steps

Figure 6.6: Contour advection with reinitialisation after varied number of time-steps

mitigated by maintaining accuracy, reinitialising every time-step.

6.3.3 Number of Reinitialisation Iterations

The final variable that will be investigated is the number of reinitialisation iterations that are undertaken between time-steps. In this test, reinitialisation is triggered after five time-steps and allowed to continue for a varying number of iterations. The results for area loss and contour skew are presented in 6.7 Parts (a) and (b).

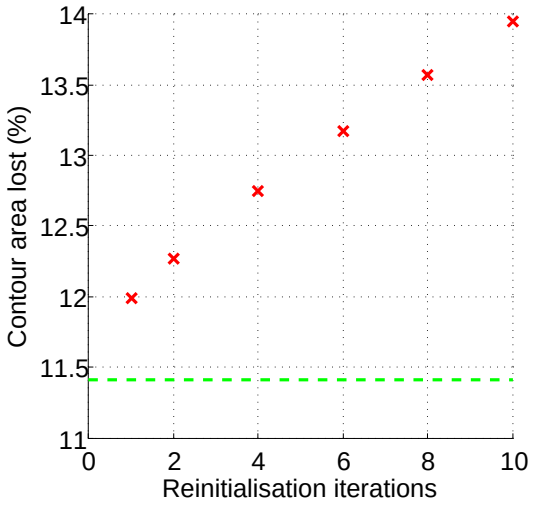
In this situation the area lost from inside the level set contour is directly proportional to the number of reinitialisation iterations. Comparing results to those of the unreinitialised result (green line) area loss is worse in all cases. This fact returns to the point made in the previous section, that if reinitialisation is not performed every time-step the contour position is moved, adversely affecting the area.

Contour skew decreases with each successive reinitialisation iteration, the exception being a single iteration. The skew value is somewhat more accurate than the result produced by two iterations per time-step. All the results are so much more accurate than the unreinitialised result, the green line does not appear within the scale on the graph.

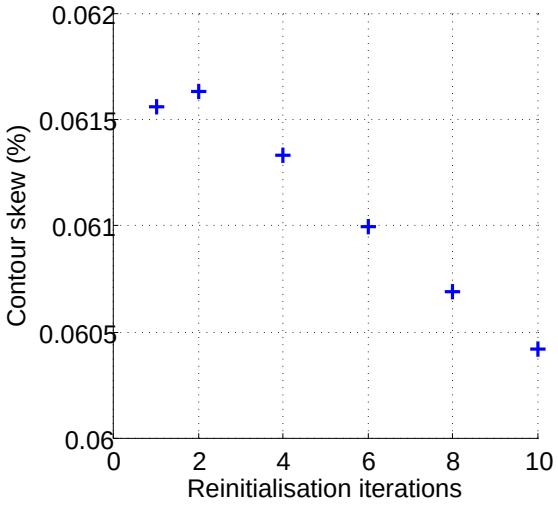
It would appear both from Figures 6.6 and 6.7, the more reinitialisation iterations that are present in the simulation, the greater area loss. Additionally, greater area loss reduces the skew of the level set contour. In this case, it is unlikely the reinitialisation procedure is improving shape accuracy and more likely that the reduction in the contour area is affecting the contour skew. The exception to this, of course, is a single reinitialisation step. By maintaining the accuracy of the signed distance property of the level set function a single iteration improves both area conservation and contour skew.

6.3.4 Concluding Remarks on Optimisation

The following tests have confirmed what was remarked upon in Sussman *et al.* (1994), that a single reinitialisation iteration per real time-step yields the most accurate results. In addition to this a pseudo time-step CFL number of around 0.1 is optimum for balancing area conservation and contour skew properties. Furthermore, it has been demonstrated that the reinitialisation procedure has the tendency to slightly move the position of the zero level set, an undesirable side effect. This factor has been mentioned previously, with Russo and Smereka (2000) suggesting a change to the method of Sussman *et al.* (1994) by altering the method of gradient approximation in the cells adjacent to the zero contour. This prevents information being taken from the incorrect side of the level set zero contour. This is not an approach that can be readily applied to the new LSL method and is therefore not in the scope of this PhD project.



(a) Area lost vs. Number of reinitialisation iterations



(b) Contour skew vs. Number of reinitialisation iterations

Figure 6.7: Contour advection varying number of reinitialisation iterations between time-steps

6.4 Best Performance Comparison of Level Set Reinitialisation

Having studied the performance of the CIP advection scheme with and without reinitialisation and the reinitialisation scheme in some detail, a final comparison demonstrating the best performance of an unreinitialised simulation and one using the novel least squares limited reinitialisation scheme is presented at resolutions of 193×193 and 385×385. Best area conservation, contour skew performance and computational time are compared to give some insight into the practical differences using a level set reinitialisation scheme could bring to a CFD analyst. The test case of the 2D cylinder advected around a square domain for three full revolutions is continued to allow for a consistent comparison throughout this Thesis.

Resolution	CFL Number	Pseudo CFL	Area Loss %	Contour Skew %	CPU Time (sec.)
Unreinitialised					
193×193	0.5	N\A	11.41%	0.1687 %	31.96
193×193	0.2	N\A	9.65 %	0.1668 %	44.51
193×193	0.1	N\A	9.06 %	0.1662 %	60.25
Reinitialised					
193×193	0.5	$0.06\dot{6}$	9.98 %	0.0221 %	41.46
193×193	0.2	$0.06\dot{6}$	5.10 %	0.0187 %	69.86

Table 6.1: The optimal results on a 193×193 grid resolution

Table 6.1 summarises the best results produced from the CIP advection solver on a 193×193 grid with and without the 2^{nd} order reinitialisation routine. Results at three different CFL numbers are given for a comparison with the reinitialised results. It should be noted that results at a CFL number of 0.1, for the time-step, are included with the unreinitialised advection solver for accuracy comparison with the reinitialised results.

One other remark to make is the absence of reinitialised results using a CFL number of 0.1. It was found that in this scenario results decreased in accuracy when compared with results at higher CFLs. This is almost certainly a result of the level set function not being advected far enough to require reinitialisation at each time step. It is likely that a successful result could be achieved if reinitialisation was reduced in frequency, however having already produced more than satisfactory results, no further investigation was completed.

The advection results without reinitialisation show that area conservation of the 2D cylinder improves as the CFL number decreases, which was already demonstrated in Section 3.9. Conversely the computation time for the problem increases due to the increased number of time-steps that require calculation.

When comparing results using reinitialisation, the best results were achieved using $0.06\dot{6}$ as the CFL for the pseudo time-step during reinitialisation. In this configuration results calculated

using a CFL number of 0.5 have a similar area preservation ability to the unreinitialised results, for a similar computational effort. However, the shape accuracy of the reinitialised results is significantly better, showing an 86.8% improvement.

A comparison of the results at a CFL number of 0.2 shows the reinitialisation method improving area conservation by 47.2% and ultimately improving area conservation by 43.7% over any of the unreinitialised advection results. This improvement is still delivered at a similar computational cost, with the shape accuracy of the the 2D cylinder further improved.

Resolution	CFL Number	Pseudo CFL	Area Loss %	Contour Skew %	CPU Time (sec.)
Unreinitialised					
385×385	0.5	N\A	5.80 %	0.0397 %	157.50
385×385	0.2	N\A	4.87 %	0.0396 %	247.63
385×385	0.1	N\A	4.56 %	0.0395 %	403.58
Reintialised					
385×385	0.5	$0.06\dot{6}$	5.04 %	0.0022 %	256.56
385×385	0.5	0.05	4.95 %	0.0022 %	251.12
385×385	0.2	$0.06\dot{6}$	2.58 %	0.0032 %	483.15
385×385	0.2	0.05	2.39 %	0.0039 %	491.25

Table 6.2: The optimal results on a 385×385 grid resolution

Continuing the comparison of results at the higher resolution of 385×385 it can be seen that the increase in resolution has the expected improvement in area conservation and a significant improvement to shape accuracy. As always, reducing the CFL number for the unreinitialised advection improves area conservation but offers almost no improvement to shape accuracy.

Comparing the results produced on the same resolution grid, it was found that a reduction in the pseudo time-step CFL for the reinitialised results from $0.06\dot{6}$ to 0.05 benefits the area preservation. At the increased grid resolution smaller re-distancing iterations are required.

The reinitialised results produced using a CFL of 0.5 are comparable in area conservation and computational effort with results produced by the unreinitialised simulation at a CFL of 0.2, however the reinitialised results show greatly improved shape accuracy. Additionally, an improvement in area conservation of 47.6% over the unreinitialised results can be achieved with a 21.7% increase in CPU time by lowering the CFL number to 0.2. The slight deterioration in shape accuracy is likely due to the very small size of the error produced and large increase in number of calculated time-steps.

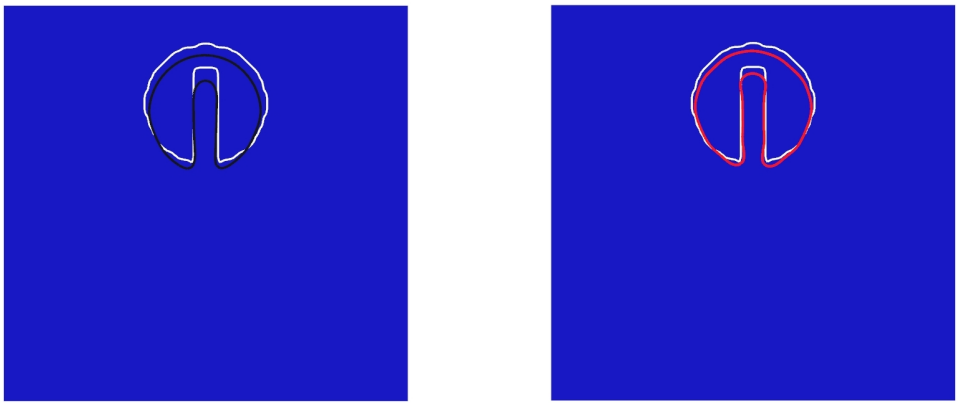
The best result produced in the comparison has a 2D cylinder advected for 3 revolutions of the domain preserving 97.61% of the original area and shape skew of only 0.0039%. Of equal importance is the fact that some of the best results produced using the reinitialised advection

solver on the lower resolution grid outperform the results in the first line of Table 6.2 and take less than 50% of the computation time.

The conclusion from this section is clear, the user can expect similar area preservation accuracy for a similar CPU time with greatly improved shape accuracy. In some cases it is possible to obtain similar accuracy at a lower grid resolution using a reinitialised solution. The consequence of which is a significantly reduced CPU time. At the other end of the spectrum, highly accurate results, obtained using a high resolution grid and time-step, can be further improved by using a reinitialisation scheme provided the corresponding increase in computation time can be afforded.

6.5 Final Test Case - Zalesak's Disc

The final test case that is presented in this Thesis is a more challenging extension to the 2D cylinder problem. The problem is known as Zalesak's Disc (Zalesak, 1979), introduced as a static re-distancing problem in Chapter 5, the full test case involves advection of the slotted disc counter-clockwise for a single revolution. The original problem was conducted on a 100×100 cell grid with a disc of 30 cells in diameter with a slot of 5×25 cells cut into the bottom of it. The centre of the 'disc' is initialised 25 cells up from the center of the domain and advected by a velocity field of $\mathbf{u} = (-2\pi y, 2\pi x)$. While the test is similar to the 2D cylinder test, the sharp discontinuous slot is extremely challenging for advection schemes to preserve. Numerical diffusion acts to cause the slot to lose sharpness and eventually disappear.



(a) No reinitialisation **(b)** LSL reinitialisation scheme

Figure 6.8: Zalesak's disc test. One revolution of a 100×100 grid

To successfully initialise the level set function to represent Zalesak's disc, the domain is initialised to a value of -0.1 with the disc geometry being initialised to a value of 0.1. To smooth the discontinuous jump the domain is statically re-distanced using 100 reinitialisation iterations. This method can be used to reinitialised any complex geometry interface easily and is recommended by Sussman *et al.* (1999).

For this test, a more qualitative analysis is taken. Assessment of performance is based purely on observation and is intended to highlight the strengths and weaknesses of the current method for the reader to bear in mind before reading Chapter 7, highlighting future work the scheme requires.

Figure 6.8 shows both unreinitialised are reinitialised results and Figure 6.9 is included as an overlaid comparison of the starting contour and two results. To produce these simulations a CFL number of 0.5 was used while the pseudo time-step CFL for reinitialisation was $0.06\dot{6}$.

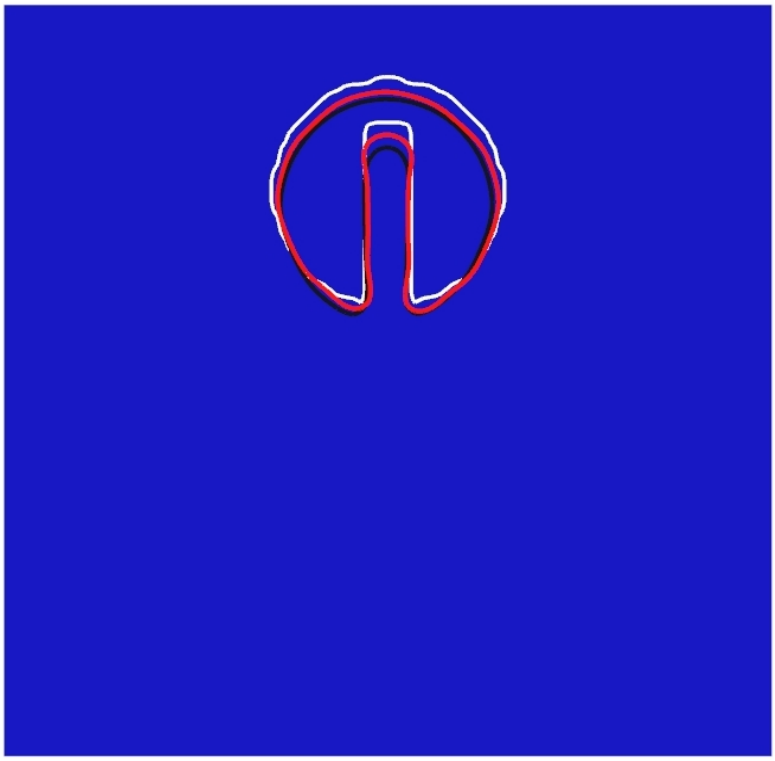


Figure 6.9: Zalesak's disc test. Comparison of unreinitialised and reinitialised results

Reinitialisation was carried out once per time-step.

In the results figures it can be seen that the new LSL reinitialisation scheme preserves the depth of the slot and the area of the disc to a better extent than the advection result without reinitialisation, confirming improved interface tracking capability. Despite the improved shape preserving performance of the reinitialisation scheme, some deformation of the interface is evident at the sharp corners of the slot. Both the base of the slot and the leading tips of the slot appear slightly distorted from the disc. This is likely to be due to some movement of the interface during the reinitialisation of the domain. Increased reinitialisation CFL number leads to more pronounced deformation at these corners. This indicates that the reinitialisation routine must be used with the appropriately chosen parameters to maximise the accuracy of results.

A step that could be taken to improve the interface preservation of the reinitialisation routine would be in the selection points used to estimate the least squares gradient. By limiting the selection points used for the least squares approximation to only those that fell in the upwind

direction, undesirable movement of the interface could be reduced.

Interfacial flows such as breaking ocean wave do not have sharp, discontinuous slots in them and this represents a great challenge to an advection solver. Accuracy on a smooth interface has already been shown to be good. Further investigation of what can be done to improve the performance of the scheme is warranted.

6.6 Concluding Remarks on Reinitialisation Methods

In this chapter it has been demonstrated that level set reinitialisation can improve the accuracy of interface tracking in the case of two test problems.

The 1^{st} order reinitialisation scheme does not improve the performance of the interface tracking over a method with no re-distancing at all. Introducing the area preserving constraint does improve the shape accuracy of the 2D cylinder with a detrimental effect on the area preservation. Regardless of the reinitialisation method, 1^{st} order spatial approximation does not provide sufficient accuracy to be acceptable for complex CFD simulations.

Using the new LSL reinitialisation scheme improves the accuracy of the results to a level above the reinitialised advection results. While the area preservation results are better using the LSL scheme adding the area preservation constraint has a negative impact bringing the figure close to that of the unreinitialised results. However it is the shape accuracy of the reinitialised results of both types that are significantly improved.

Using the most successful reinitialisation method, testing showed that the signed distance of the level set function is best maintained every time-step with only a small correction requiring a single reinitialisation iteration. Increasing the number of reinitialisations used between time-steps failed to improve the performance of the scheme.

Considering the reinitialised level set scheme against the unreinitialised results, it was shown that for some increase in computational expense, the area preservation could be improved with shape the accuracy improved significantly. In some cases, a similar accuracy could be achieved using the reinitialisation scheme on a lower resolution grid, resulting in a reduction in computational time. For those requiring the highest levels of accuracy, high resolution results could be improved further by using the reinitialisation scheme.

Obtaining results from a different test problem, Zalesak's disc, it was shown that the reinitialisation scheme could be used to improve the preservation of the discontinuous slot feature of the shape. While the slot was preserved to a greater degree than that of the unreinitialised result, there was slight deformation of the shape at the sharp corners of the geometry, indicating some undesirable movement of the level set contour. The possibility of improving the scheme exists if points only lying in the upwind area of the level set function were sampled for use in the least squares approximation.

Chapter 7

Conclusions and Future Work

This thesis presents a description of a novel least squares limited level set reinitialisation method for the first time. The demand for accurate tracking of fluid interfaces in segregated flow problems, such as breaking ocean waves, is strong. While Volume of Fluid (VOF) methods have been popular for simulating multi-fluid flow problems, their discontinuous representation a fluid's free surface is unrealistic and causes difficulty when applying free surface boundary effects such as surface tension (Scardovelli and Zaleski, 1999). The level set method (Osher and Sethian, 1988) is an alternative to VOF schemes, who's free surface representation is implicit, continuous and smooth, ideally suited to this class of problem.

Due to the need for the level set function to represent a signed distance from the interface, the function must periodically be 're-distanced'. One of the most efficient manners in which this can be achieved is through the process of reinitialisation Sussman *et al.* (1994); Sussman and Fatemi (1999). While the reinitialisation procedure is commonly carried out on structured grids, there is a need for reinitialisation to be implemented for other types of grid. Of particular interest was the hybrid grid type, Cartesian cut-cell (Ingram *et al.*, 2003). Its ability to represent complex geometries in a simple manner, without effecting the quality of the entire grid is attractive. To facilitate the use of level set reinitialisation on these types of grid, a new gradient approximation scheme was developed, capable of extension to both structured and unstructured grid types.

7.1 General Summary

The objectives outlined in Section 2.5 were successfully met. A new method of gradient approximation has been developed for use with level set reinitialisation methods (Sussman *et al.*, 1994, 1998; Sussman and Fatemi, 1999). The use of a new gradient approximation method allows for future extension to unstructured grids. This facilitates the efficient, accurate and flexible simulation of free surface flows such as breaking ocean waves.

A CIP solver was implemented with the purpose of transporting a level set function about a domain for testing and evaluation. With sufficient grid resolution established through a grid

convergence study. A level set reinitialisation method by Sussman *et al.* (1994) was selected as an appropriate method by which to maintain an accurate signed distance function, necessary for successful interface tracking. Additional accuracy was added using an area preserving constraint developed by Sussman *et al.* (1998); Sussman and Fatemi (1999).

A level set reinitialisation routine that was capable of operation on structured or unstructured grids of mixed geometry was required. A suitable method was designed using a least squares approximation and a slope limiter to guarantee the stability of the reinitialisation process. The method was successfully tested on a structured grid with description of how it should be implemented on an unstructured grid. The results of the method showed an improved accuracy when tracking the interface of two test problems.

7.1.1 CIP Advection Solver

Before a level set method could be used to track an interface, an appropriate solver was required to perform the advection of the signed distance function. The Cubic Interpolated Polynomial method Yabe and Aoki (1991) was chosen as a efficient, stable method that was suitable for use with level set methods. Watanabe *et al.* (2005) already had success using this method for simulating breaking waves on a sloping beach.

CIP 0 and 1 advection solvers by Yabe and Aoki (1991) were successfully implemented and tested in 1D. A qualitative analysis of these results determined that the CIP 1 method offered little advantage over CIP 0 when acting as an advection solver for level set method. This was because the level set function's smooth monotonic nature guaranteed that there were no areas of discontinous data removing the need for the CIP 1 method to suppress Gibbs phenomenon.

The CIP 0 method was extended to 2D (Yabe *et al.*, 1991) and a grid convergence study was conducted to verify the implementation of the CIP method, and to determine suitable resolutions for the testing of the accuracy of the level set method for the remainder of the project. A circular contour advection test was chosen to provide a smooth interface whose area and shape could easily be compared between the start and end time-steps.

7.1.2 Level Set Method and Reinitialisation

A level set reinitialisation routine by Sussman *et al.* (1994) was implemented for the purpose of maintaining the accuracy of the signed distance property of the level set function between simulation time-steps. A series of 1D tests were conducted, demonstrating the successful redistancing of smooth and discontinuous test functions both statically and with advection. The method was extended for use in 2D simulation using 1^{st} order upwind approximation. Static tests with the CIP advection solver from the previous chapter were repeated. Due to the order of accuracy of the gradient approximation, the reinitialisation method made no improvement on the advection test results.

To improve the performance of the reinitialisation routine, an improved method by Sussman and Fatemi (1999) was trialled. It included an area conserving constraint that prevented the movement of the zero level set contour during reinitialisation. The addition of the constraint helped to increase the accuracy of the level set reinitialisation. While the new method improved the shape of the contour significantly, the area conservation of the method was decreased. Therefore no decision on a preferred reinitialisation routine was made at that point.

7.1.3 A New Least Squares Limited Reinitialisation Method

In order to use a reinitialisation routine accurately on an unstructured grid type, the 1^{st} order spatial approximation used in the reinitialisation procedure had to be changed to a higher order approximation method that was suitable for use on such grids. To fulfil this requirement a least squares gradient approximation was trialled. However due to the nature of the Hamilton-Jacobi equation used to re-distance the level set function, the least squares approximation was limited to 1^{st} order accuracy when an over-estimate of the gradient was detected. This guaranteed monotonicity and the stability of the re-distancing procedure.

Static re-distancing tests demonstrated a successfully stable reinitialisation scheme. However, an unexpected side-effect of the method was a numerical artefact that occurred in the form of a reinitialisation spike in the level set function. The spike was found to occur at turning points points where the gradient of the level set function was indeterminate. To prevent the spikes a gradient sign test was used to determine the location of turning points and least squares approximations were replaced with upwind approximations where appropriate. Because the reinitialisation spikes occurred at function maxima and minima, they were found outwith the location of the important interface and did not have a significant effect on the accuracy of the interface tracking scheme.

7.1.4 Extension of the New Reinitialisation Procedure to Unstructured Grids

An initial attempt to extend the reinitialisation method to unstructured grids using data from approximate local points did not produce satisfactory results. In the worst case scenario that was tested, the slope limiter acted to remove a component of gradient in each axis direction, distorting the signed distance function and resulting in a level set function with gradients aligned to the major axes. This work did, however, provide an insight into how the reinitialisation method should be extended to other grid types and suggestions for further extension work were made.

Having established the limitations of the slope limiter an alternative method was suggested to extend the reinitialisation routine to unstructured grids. To provide more accurate data with which to implement the slope limiter, this method involved reconstructing the level set method at a new set of grid points, lying on the level set normal at each grid point. Interpolation of data is a practice that has been widely used by others. This approach represents the next stage

of research for continuation of the project. The development required to do so is discussed in Section 7.2.

7.1.5 Results and Performance of the New Reinitialisation Method

Testing of the successful LSL reinitialisation method using the circular contour advection test showed that the method could track an interface more accurately than an unreinitialised level set method or the 1st order reinitialisation method. Using the LSL method there was a small improvement in area conservation but the greatest benefit was found in the shape accuracy of the contour. The inclusion of the area-conserving constraint was found not to significantly improve the result when used with the new gradient approximation scheme.

In an optimisation exercise, it was shown that, for the particular test case, the optimal use of the reinitialisation procedure was to use a short pseudo-time step for a single reinitialisation iteration between each simulation time-step. This maximised the accuracy of the signed distance function and minimised the error in area conservation. These findings matched closely with what was reported in Sussman *et al.* (1994).

In a further exercise it was shown that results of a similar accuracy could be obtained using a grid at half resolution and a reinitialisation for a much lower computational cost. If the user was not limited by computational time, the accuracy of results used on a higher grid resolution far exceeded the results obtained without reinitialisation.

Finally a test using Zalesak's disc (Zalesak, 1979) was conducted. A qualitative comparison between unreinitialised advection and the new reinitialisation scheme showed improved performance in shape and area preservation. In particular, the depth of the slot in the disc was maintained far more effectively when reinitialisation was implemented. Some distortion of the the sharp corners of the disc was apparent but the overall reinitialised result was a closer match to the original un-advected shape.

The reinitialisation method presented in this Thesis has sufficient promise and development scope to warrant further investigation, with improvements in both accuracy and grid generality. These improvements could be implemented using the suggestions for future work described in the following section.

7.2 Future Work

There are a number of improvements and areas of further investigation that could be incorporated into future research following on from this project. Based on the work presented in this Thesis there are opportunities to improve the level set method's accuracy, generality and computational performance.

Of greatest importance is the implementation of the least squares limited reinitialisation scheme on a Cartesian cut cell or unstructured grid. To achieve this, the gradient approximation scheme and reinitialisation routine must be adapted to account for points at arbitrary locations. The reinitialisation scheme itself, should require little modification, as it only requires the correctly approximated Godunov Hamiltonian at each grid point to perform the re-distancing procedure. The least squares limited scheme must be implemented in the manner outlined in Section 5.6.2.

Before updating the reinitialisation method to work on a different grid type, a suitable grid code must be found. The porting and implementation of existing methods to a new grid framework is potentially straightforward but time consuming. This was a principle factor in ending the development work of the project at its current point. In the case of developing the reinitialisation method for Cartesian cut cell grids, the existing method could be implemented immediately on areas of structured grid. An interpolated slope limiting method must be implemented on zones of cut cells with testing to assess the method's effectiveness. In the case of an unstructured grid, the interpolation method would be required passing over all cells.

Aside from the slope limiter, there are potential improvements to be made to the least squares approximation. As Russo and Smereka (2000) discusses, the inclusion of points from the incorrect side of the level set interface can have an effect on moving the zero contour during the re-distancing procedure, affecting the accuracy of the method. Further to this the re-distancing method 'propagates' from the zero contour position, re-distancing the domain in a 'wave'. It is technically invalid to include points from the downwind direction. To this end, a possible method of improving the least squares approximation is to experiment with a weighted approximation, such as the method presented by Mandal and Subramanian (2008). It could be possible to choose the relevant weights to the approximation points depending on their 'upwindedness' in the solution. Further to this it may be possible to entirely exclude downwind points from the least squares stencil, redefining it to only include points from the upwind zone of the level set function.

There are additional sources of improvement for the reinitialisation method, outside of the spatial approximation. it is clear that the temporal accuracy could be improved. Although the reinitialisation method is called between simulation time-steps, the method includes a pseudo-time quantity. Therefore, a higher order explicit form of time integration method can be implemented to further improve the re-distancing accuracy. The most popular high order time integration methods are the multi-step Runge Kutta methods. A 2^{nd} order Runge Kutta scheme

is suggested for reinitialisation by Sussman *et al.* (1994); Sussman and Fatemi (1999). Further effort in implementing higher order methods for the temporal scheme is likely to yield too small a gain for additional computational cost.

When further considering an implementation on Cartesian cut cell grids, there is another step that could be completed to further improve the accuracy of the reinitialisation method. A number of authors, such as Kim and Liou (2007), have hybridised the spatial approximation on adaptive grids, using a high order WENO5 method for the regular parts of the mesh and a 2^{nd} order scheme at points where insufficient stencil points exist, around the hanging nodes of the grid. This presents a similar opportunity when considering the limited least squares method. It could be possible to hybridise a high order ENO scheme, replacing it with the LSL method in areas where the ENO stencil did not work. These areas could be in non-uniform cells, at hanging nodes (if an adaptive mesh were implemented) and at boundaries of problems where large stencil ENO schemes can be difficult to implement.

To improve the efficiency of the level set method a number of authors have suggested methods designed to limit its computational requirements. Many of these schemes are 'narrow band' schemes, as the level set method need only be accurately know in a region about the interface it is tracking. The far field values are irrelevant to helping track the interface. Narrow band methods have been suggested by authors such as Adalsteinsson and Sethian (1995), however there are difficulties with the treatment of the level set values at the edge of the band. One such method that appears to successfully address the issues of narrow band methods is that of Peng *et al.* (1999). Based on the methods presented in this thesis there is no obvious obstacle to implementing this type of narrow band approach. Such a method would help to reduce memory use and increase computational efficiency of simulations using the level set method.

Finally, the CIP method implemented in this PhD project is based on the finite difference method. To realise the potential of the methods developed here a move to a more flexible finite volume method is necessary. One such method that is worthy of consideration is a finite volume implementation of the CIP method by Ii and Xiao (2007). This method is still based on construction of a constrained polynomial function, however applied in a finite volume formulation using a TVB spatial approximation. Work to implement this method is currently being undertaken at The Unviersity of Edinburgh's Institute for Energy Systems.

Appendix A

Appendix B - A Comment on FORTRAN Programming and Software Design

Some comments about the software programming challenges faced during the project...

By far one of the greatest challenges I faced during my PhD project, but also a great motivation, was that of learning the software development skills with which to be able to implement the algorithms presented throughout this Thesis. Coming from a mechanical engineering background I was somewhat naïve to the demands of writing, debugging and maintaining a code base. Before starting my PhD experience, I had spent some time 'programming' in applications such as Matlab and using VBA for Microsoft Excel. Unfortunately, none of my efforts had ever extended single module programs and some form of run time debugging was normally available.

From an early stage in the project it was decided that I would develop my own code base from scratch. There were several reasons for this. The drivers were the lack of existing code relevant to my subject area held by the Institute for Energy Systems and the chance to learn to implement mathematical algorithms from descriptions first principles. This decision undoubtedly was responsible for consuming much of my time in the early years of my project but it is one that I am glad I stuck to. The alternatives presented their own difficulties.

Within my first year of studies I assessed the feasibility of using the highly popular open source CFD library, OpenFOAM. Written in C++ and containing a huge number of classes, OpenFOAM provides everything that the budding numericist could ask for. However with such a large code base comes a very long learning curve. Given that I had no prior knowledge of C++ programming, or indeed OO programming principles, it was decided that the risk associated with undertaking working with this code base was too great. Even now, having completed my PhD project and working as a professional software developer, a significant amount of effort would be required to implement the techniques in this thesis using OpenFOAM. The other option was to find some code that had been released by other research groups. How-

ever, this presents its additional difficulties, requiring a lengthy code reviews and verification. More importantly, some of the research code I was offered was extremely poorly commented, monolithic and uses variable names translated from Japanese! I think in the end I was justified spending some time constructing my own code library.

I started my development work in Linux using a mixture of development tools. Having written some short test programs and examples, I quickly realised that close management of software projects was extremely important. I very shortly afterwards embraced the security of managing my code base using Subversion. Having seen some of my supervisor's code I was keen to build a modular code base, sharing as many of my subroutines and functions between programs as possible. To this end, my library has been very successful. I have maintained separate code bases for 1D and 2D methods with a small number of shared modules. This undoubtedly increased the efficiency of my work. Further to this I have found that modular code is essential for verification and testing purposes. Debugging modules of code in small example programs a few hundred lines long at a time is a focused manner in which to approach testing. One of the first programs I wrote did not follow this approach. I wrote a 1D reinitialisation test program, unplanned from start to finish with no testing until it was complete. As I remember it took me about 2 months of effort to debug and verify my code correctly. Needless to say I was very 'green' at the time.

Another lesson I learnt early on was that a software developer's life is immediately 100 times more difficult if a programming project is started without a thoroughly planned and detailed program design. Program design not only helps you write clear concise code that is easily understandable and maintainable by others but it also ensures the your total understanding of the problem before implementation, thereby reducing the likelihood of software bugs by a considerable margin.

I have always tried to write code with some performance optimisation in mind. My primary goals were to attempt to access memory in an efficient manner, reading through arrays in sequential order; limit the use of logic statements where ever possible and use of loops that were easily vectorisable by the compiler. Each of these steps gave some tangible speed increases as I leaned and implemented them. Some more advanced programming concepts were omitted as I did not take advantage of the use of memory pointers were not generally taken advantage of. This inevitably has resulted in a large number of memory copies that could be removed in the future. However as a novice programmer the importance of being able to produce stable bug free computer code cannot be underestimated. My code has certainly been relatively easy to maintain and understand (if not debug) as I have grown my code library over 3 years.

Appendix B

Appendix A - An Algorithm for Measuring Contour Area and Perimeter

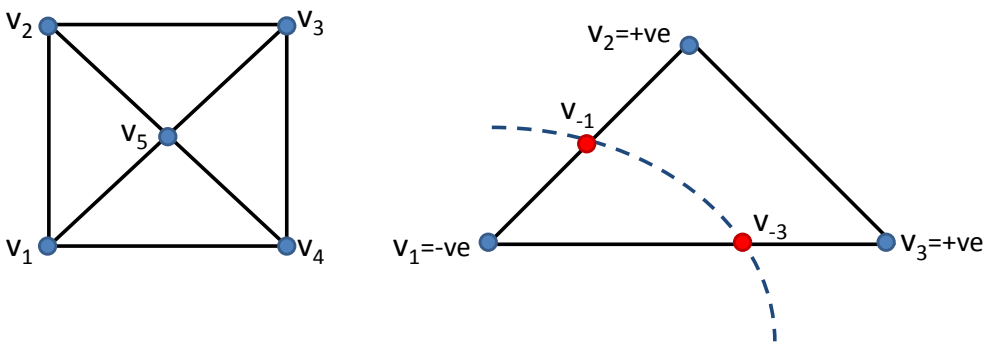


Figure B.1: Cell decomposition and sub-cell triangle vertex nomenclature

The following FORTRAN module was used for analysis purposes during my PhD project. It was one of the first FORTRAN programs I wrote at the start of my studies and has been one of the most useful. I cannot take the credit as architect of the algorithm, I was much help by my supervisor. The contribution of the rather lovely and highly efficient 'bit flipping' lookup table (explained below) is from an unknown source, passed on to me by my supervisor.

The purpose of this algorithm is to measure the area defined by either a positive or negative level set function. For simplicity, the algorithm is described below in pseudo-code for calculating the area and perimeter of the negative signed level set on a regular Cartesian grid.

Figure B.1 shows how the individual cells are decomposed into triangles and the nomenclature of the vertices when a zero level set contour cuts divides a cell triangle. If all the vertices of a triangle are positive then the triangle is discounted, however there are seven ways a triangle can contain some area of negative level set. Using the nomenclature displayed in Figure B.1 these are summarised.

Point *bits*			No. vertices	Vertex A	Vertex B	Vertex C	Vertex D
1	2	3					
0	0	0	3	1	2	3	0
0	0	1	4	-1	2	3	-3
0	1	0	4	1	-1	-2	3
0	1	1	3	-3	-2	3	0
1	0	0	4	1	2	-2	-3
1	0	1	3	-1	2	-2	0
1	1	0	3	3	1	-3	0

Table B.1: Cell vertex lookup table

The special property of this table is that based on the values of the three vertices, a 3-bit number can be used to represent the condition of the cell. Using a bit flipping function, *ibset*, intrinsic to FORTRAN 95 a single check on each vertex can be used to generate the outcome represented in the table above. The appropriate linear interpolations can be performed the generate the values at the intermediate points of v_{-1}, v_{-2}, v_{-3}

The main area calculation module is included below for lack of a more concise and clear manner of explaining the algorithm. The main routine and short module used for reading input data are omitted due to their trivial nature.

```
module AreaCalculations
  implicit none
  !The lookup table contains information about how may vertexes
  !the negative area polygon contains, and which vertexes these
  !are made of. Column 1 is number of vertexes - 3 or 4.
  !Columns 2 through 4 are the row numbers of TriVertex and the
  !final column is a repeat of the first column. (do loop)
  integer, dimension(6,7), parameter:: VertexCheck &
       &=reshape((/3, 1, 2, 3, 1, 0, &
       &4, 5, 2, 3, 7, 5, &
       &4, 1, 5, 6, 3, 1, &
       &3, 7, 6, 3, 7, 0, &
       &4, 1, 2, 6, 7, 1, &
       &3, 5, 2, 6, 5, 0, &
       &3, 1, 5, 7, 1, 0/),shape=(/6,7/))
  !eps divide by 0 prevention
  real(kind=dp), parameter :: eps = epsilon(eps)
contains
```

```
subroutine GridAreaCalc(Area, PerimeterLength, Xmin, Ymin, &
                        Xmax, Ymax, XCells, YCells, NodeData)
  ! Passed variables
  real(kind=dp):: Area
  real(kind=dp):: PerimeterLength
  real(kind=dp):: Xmin, Ymin, Xmax, Ymax
  real(kind=dp), dimension (:,:):: NodeData
  integer :: XCells, YCells
  !Descriptions: Xmax/Ymax are the Maximum X and Y values of
  !the grid respectively. Xcells/Ycells are the number of
  !cells that occur in the X and Y directions. Xlen/Ylen
  !dimensions of a cell. [To be modified to accomodate a
  !graded mesh] StepArea, PolyArea, CellArea and GridArea
  !are the calcualted areas for each loop of the program.

  ! Local variables
  real(kind=dp), dimension(6):: CellScalar
  !Description: CellScalar is a column array of scalar values
  !for each vertex of a cell. Entry 5 is a repetition of entry
  !1 as a short cut in do loops and entry 6 is the value at
  !the centre of the cell.

  real(kind=dp), dimension(2,6)::CellVertex
  !Description: CellVertex is a 2 X 6 array of vertex
  !coordinates describing the vertexes of the cell.
  !Coordinate 5 is a repeat of 1 and coordinate 6 is the
  !center

  real(kind=dp):: Xlen, Ylen, StepArea, CellArea, eps
  integer :: i,j,k

  !Calculate the coordinates of the vertexes of the cell
  Xlen=(Xmax–Xmin)/(Xcells–1)
  Ylen=(Ymax–Ymin)/(Ycells–1)

  ! loop over all the cells in the grid.
  do i=1,Xcells–1
     do j=1,Ycells–1
        CellVertex(1,1)=Xmin+real(i–1)*Xlen
```

```
            CellVertex(2,1)=Ymin+real(j-1)*Ylen
            CellVertex(1,2)=Xmin+real(i)*Xlen
            CellVertex(2,2)=Ymin+real(j-1)*Ylen
            CellVertex(1,3)=Xmin+real(i)*Xlen
            CellVertex(2,3)=Ymin+real(j)*Ylen
            CellVertex(1,4)=Xmin+real(i-1)*Xlen
            CellVertex(2,4)=Ymin+real(j)*Ylen

            ! install the vertex points into cell scalar
            CellScalar(1)=NodeData(i,j)
            CellScalar(2)=NodeData(i+1,j)
            CellScalar(3)=NodeData(i+1,j+1)
            CellScalar(4)=NodeData(i,j+1)

            !Check all cell corner values. All +ve cells are
            !discarded. All -ve - full cell area is included
            !in the area calculation.
            if(any(CellScalar(1:4)<0.0_dp))then
               !add in short cut for do loop
               CellVertex(:,5)=CellVertex(:,1)
               CellScalar(5)=CellScalar(1)

               !Calculate the X/Y coords for center of the cell
               CellVertex(1,6)=Xmin+Xlen/2.0+real(i-1)*Xlen
               CellVertex(2,6)=Ymin+Ylen/2.0+real(j-1)*Ylen

               !...and its scalar value
               CellScalar(6)=(CellScalar(1)+CellScalar(2)+ &
                             &CellScalar(3)+CellScalar(4))/4

               !Calculate Cellarea
               call CellAreaCalc(CellArea, PerimeterLength, &
                                 &CellVertex, CellScalar)
               !and add it to the grand total.
               Area=Area+CellArea
            end if
         end do
      end do
   end subroutine GridAreaCalc
```

```
subroutine CellAreaCalc(CellArea, PerimeterLength, &
                        &CellVertex, CellScalar)
  !Passed variables
  real(kind=dp):: CellArea
  real(kind=dp):: PerimeterLength

  real(kind=dp), dimension(6):: CellScalar
  !Description: CellScalar is a column array of scalar
  !values for each vertex of a cell. Entry 5 is a repetition
  !of entry 1 as a short cut in do loops and entry 6 is the
  !value at the centre of the cell.

  real(kind=dp), dimension(2,6)::CellVertex
  !Description: CellVertex is a 2 X 6 array of vertex
  !coordinates describing the vertexes of the cell.
  !Coordinate 5 is a repeat of 1 and coordinate 6 is the
  !center

  !Local Variables
  real(kind=dp), dimension(4):: TriScalar
  !Description: TriScalar is a column array of scalar values
  !for the current cell triangle being calculated. Entry 4
  !is a repetion of 1 as a do loop short cut.

  real(kind=dp), dimension(3):: Alpha
  !Description: Alpha is an array of fractions that
  !determines when the zero point occurs on a triangle edge.

  real(kind=dp), dimension(2,7)::TriVertex
  !Description: TriVertex is a 2 X 7 array of vertex
  !cooridinates describing the current triangle being
  !processed. Coordinates 1-3 are the 3 vertexes.
  !Coordinate 4 is a repear of coordinate 1. Coordinates 5-7
  !are the positions of interpolated points along the edges
  !of the triangle.
```

```
real(kind=dp):: PolyArea

!Area calculated for a given cell triangle
integer:: CellNumber, CellTriangle, VertexNum, &
          &VertexIndex, InterpolNum, PolyNum, i
!Descriptions: All counting integers for do loops.

CellArea=0.0
TriVertex=0.0

do CellTriangle=1, 4
   !Assign the TriVertex values from CellVertex values for
   !the calculation.
   TriVertex(:,1)=CellVertex(:,CellTriangle)
   TriVertex(:,2)=CellVertex(:,CellTriangle+1)
   TriVertex(:,3)=CellVertex(:,6)
   TriVertex(:,4)=TriVertex(:,1) !Shortcut

   !Assign the correct scalar values for corners 1:3
   TriScalar(1)=CellScalar(CellTriangle)
   TriScalar(2)=CellScalar(CellTriangle+1)
   TriScalar(3)=CellScalar(6)
   TriScalar(4)=TriScalar(1) ! shortcut

   !Find the vertex index for triangle
   !A count of 1 indicates >= 0 value vertexes
   VertexNum=0
   do VertexIndex = 1,3
      if (TriScalar(VertexIndex)>=0) VertexNum= &
           & ibset(VertexNum, VertexIndex-1)
   end do

   !discount any triangles with all positive scalar values
   if (VertexNum==7)cycle

   !Interpolate edges to calculate polygon coordinates
   !Calculate the fraction along the edge that the 0 point
   !occurs.
   do InterpolNum=1,3
```

```
            Alpha(InterpolNum)=abs(TriScalar(InterpolNum))/ &
                 & (abs(TriScalar(InterpolNum)- &
                 & TriScalar(InterpolNum+1)+eps))

            !Then use x=x0+Alpha(x1-x0)to calc x coordinate and
            !similar for y.
            TriVertex(1,4+InterpolNum)=TriVertex(1,InterpolNum)+&
               & Alpha(InterpolNum)*(TriVertex(1,InterpolNum+1)-&
               & TriVertex(1,InterpolNum))
            TriVertex(2,4+InterpolNum)=TriVertex(2,InterpolNum)+&
                 Alpha(InterpolNum)*(TriVertex(2,InterpolNum+1)-&
                 & TriVertex(2,InterpolNum))
         end do

         !Area from each triangle summed to CellArea
         PolyArea=PolyArea(TriVertex,VertexNum+1)
         CellArea=CellArea+PolyArea

         !Calculate Perimeter of contour
         PerimeterLength=PerimeterLength+ &
            & Perimeter(TriVertex, VertexNum+1)
      end do
   end subroutine CellAreaCalc


   real function PolyArea(TriVertex,VertexNum)
      !function to calculate triangle area using vectors - signed
      !area algorithm
      !Area= sigma(1,n)1/2(x(i)y(i+1)-x(i+1)y(i))

      integer :: VertexNum,PolyNum
      real(kind=dp), dimension(2,7)::TriVertex
      real(kind=dp):: StepArea
      !Description: TriVertex is a 2 X 7 array of vertex
      !cooridinates describing the current triangle being
      !processed. Coordinates 1-3 are the 3 vertexes. Coordinate
      !4 is a repear of coordinate 1. Coordinates 5-7 are the
      !positions of interpolated points along the edges of the
      !triangle.
```

```
    !Add 1 to VertexNum so that value 0 refers to line 1 of the
    !VertexCheck array and so on
    PolyArea=0.0

    do PolyNum= 1,VertexCheck(1,VertexNum)
       !Area calc by each vertex
       StepArea=0.5*(TriVertex(1,VertexCheck &
            & (PolyNum+1,VertexNum))* &
            & TriVertex(2,VertexCheck(PolyNum+2,VertexNum))- &
            & TriVertex(1,VertexCheck(PolyNum+2,VertexNum))* &
            & TriVertex(2,VertexCheck(PolyNum+1,VertexNum)))
       !Polygon area
       PolyArea=PolyArea+StepArea
    end do
  end function PolyArea
```

Bibliography

D. Adalsteinsson and J. A. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118:269–277, 1995.

W. Allsop, T. Bruce, D. Causon, D. Ingram, C. Mingham, J. Pearson, S. Richardson, and J. Zang. Violent overtopping of waves at seawalls. Website, Last visited June 2011. URL `http://www.vows.ac.uk/`.

A. Amsden and F. Harlow. A simplified MAC technique for incompressible flow calculations. *Journal of Computational Physics*, 6:322–325, 1970.

J. Anderson. *Computational Fluid Dynamics: The Basics With Applications*. McGraw-Hill International Editions, 1995.

N. Ashgriz and J. Y. Poo. FLAIR: Flux line-segment model for advection and interface reconstruction. *Journal of Computatuional Physics*, 93(2):449–68, April 1991.

R. Ausas, E. Dari, and G. Buscaglia. A geometric mass preserving redistancing scheme for the level set function. *International Journal for Numerical Methods in Fluids*, 2010.

T. Barth and D. C. Jesperson. The design and application of upwind schemes on unstructured meshes. In *Aerospace Sciences Meeting, 27th, Reno, NV, Jan 9–12*, 1989.

T. Barth and J. A. Sethian. Numerical schemes for the Hamilton-Jacobi and level set equations on triangulated domains. *Journal of Computational Physics*, 145:1–40, 1998.

M. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics*, 82:64–84, 1989.

M. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial difference equations. *Journal of Computational Physics*, 53:484–512, 1984.

D. Book, J. Boris, and K. Hain. Flux-corrected transport II: Generalizations of the method. *Journal of Computational Physics*, 18:248–283, 1975.

J. Boris and D. Book. Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works. *Journal of Computational Physics*, 11:38–69, 1973.

J. Boris and D. Book. Flux-corrected transport. III. Minimal-error FCT algorithms. *Journal of Computational Physics*, 20:397–431, 1976.

J. Boris, A. Landsberg, E. Oran, and J. Gardner. LCPFCT – A flux-corrected transport algorithm for solving generalized continuity equations. NRL Memorandum Report 6410-93-7192, U.S. Naval Research Laboratory, 1993.

A. Bowyer. Computing Dirichlet tessellations. *The Computer Journal*, 24(2):162–166, 1981.

D. Causon, D. Ingram, C. Mingham, G. Yang, and R. Pearson. Calculation of shallow water flows using a Cartesian cut cell approach. *Advances in Water Resources*, 23:545–562, 2000.

D. Causon, D. Ingram, and C. Mingham. A Cartesian cut cell method for shallow water flows with moving boundaries. *Advances in Water Resources*, 24:899–911, 2001.

Y. C. Chang, T. Y. Hou, B. Merriman, and S. J. Osher. A level set formulation of Eulerian interface capturing methods for incompressible fluid flows. *Journal of Computational Physics*, 124:449–464, 1996.

C. Cho and S. Kim. An essentially non-oscillatory Crank-Nicholson procedure for incompressible Navier-Stokes equations. *International Journal for Numerical Methods in Fluids*, 56:1351–1357, 2008.

D. Chopp. Computing minimal surfaces via level set curvature flow. *Journal of Computational Physics*, 106:77–91, 1993.

W. J. Coirier and K. G. Powell. An accuracy assessment of Cartesian-mesh approaches for the Euler equations. *Journal of Computational Physics*, 117:121–131, 1995.

R. Courant, K. Friedrichs, and H. Lewy. On the partial difference equations of mathematical physics. *Mathematische Annalen*, 100:32–74, 1928.

M. S. Darwish and F. Moukalled. TVD schemes for unstructured grids. *International Journal of Heat and Mass Transfer*, 46:599–611, 2003.

D. De Zeeuw and K. Powell. An adaptively refined Cartesian mesh solver for the Euler equations. *Journal of Computational Physics*, 104:56–68, 1993.

J. Douglas, J. Gasiorek, and J. Swaffield. *Fluid Mechanics*. Prentice Hall, 4th edition, 2001.

D. Enright, R. Fedkiw, J. Ferziger, and I. Mitchell. A hybrid particle level set method for improved interface capturing. *Journal of Computational Physics*, 183:83–116, 2002.

D. Enright, D. Nguyen, F. Gibou, and R. Fedkiw. Using the particle level set method and a second order accurate pressure boundary condition for free surface flows. In *4th ASME JSME Joint Fluids Engineering Conferecnce, July 6-11, Honolulu, Hawaii, USA*, 2003.

M. W. Evans and F. H. Harlow. The particle-in-cell method for hydrodynamic calculations.

Technical report, Los Alamos Scientific Laboratory, 1957.

R. Fernando, editor. *GPU Gems*. Addison-Wesley, 2004.

N. Frink, P. S., and P. Parikh. An unstructured-grid software system for solving complex aerodynamic problems. *NASA Conference Publications*, 3291:289, 1995.

F. Gao, D. Ingram, D. M. Causon, and C. Mingham. The development of a Cartesian cut cell method for incompressible viscous flows. *International Journal for Numerical Methods in Fluids*, 54(9):1033–1053, 2006.

S. Godunov. A difference scheme for numerical computation of discontinuous solution of hydrodynamic equations. *Sbornik: Mathematics*, 47:271–306, 1959.

F. Harlow and J. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The Physics of Fluids*, 13:2182–2189, 1965a.

F. H. Harlow, J. P. Shannon, and J. E. Welch. Liquid waves by computer. *Science*, 149(3688): 1092–1093, September 1965.

F. Harlow. Hydrodynamic problems involving large fluid distortions. *Journal of the Association for Computing Machinery*, 4(2):137–142, 1957.

F. Harlow and J. E. Welch. Numerical study of large amplitude free surface motions. *Physics of Fluids*, 9:842, 1965b.

A. Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 49:357–393, 1983.

A. Harten. ENO schemes with subcell resolution. *Journal of Computational Physics*, 83:148–184, 1989.

A. Harten, B. Engquist, S. Osher, and S. Chakravarthy. Uniformly high order accurate essentially non-oscillatory schemes, III. *Journal of Computational Physics*, 71:231–303, 1987.

D. Hartmann, M. Meinke, and W. Schroder. The constrained reinitialisation equation for level set methods. *Journal of Computational Physics*, 229:1514–1535, 2010.

C. Hirsch. *Numerical Computation of Internal and External Flows. The Fundamentals of Computational Fluid Dynamics*. Butterworth-Heinemann, 2007.

C. Hirt and B. Nichols. Volume of fluid (VoF) methods for dynamics of free boundaries. *Journal of Computational Physics*, 39:201–225, 1981.

K. Hu, C. Mingham, and D. Causon. Numerical simulation of wave overtopping of coastal structures using the non-linear shallow water equations. *Coastal Engineering*, 41:433–465,

2000.

S. Ii and F. Xiao. CIP/ multi-moment finite volume method for Euler equations: A semi-Lagrangian characteristic forumulation. *Journal of Computational Physics*, 222:849–871, 2007.

D. Ingram, D. Causon, and C. Mingham. Developments in Cartesian cut cell methods. *Mathematics and Computers in Simulation*, 61:561–572, 2003.

D. Ingram, F. Gao, D. Causon, C. Mingham, and P. Troch. Numerical investigations of wave overtopping at costal structures. *Costal Engineering*, 56:190–202, 2009.

A. Jameson. Artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid convergence in transonic and hypersonic flows. In *11^{th} AIAA Compuational Fluid Dynamics Conference*, 1993.

H. Jasak, H. Weller, and A. Gosman. High resolution NVD differencing scheme for arbitrarily unstructured meshes. *Numerical Methods in Fluids*, 31(2):431–449, 1999.

G. S. Jiang and C. W. Shu. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics*, 126:202–228, 1996.

G. Jiang and D. Peng. Weighted ENO schemes for Hamilton-Jacobi equations. *Journal of Scientific Computing*, 21(6):2126–2143, 2000.

H. Kim and M. Liou. Three dimensional cut-cell Cartesian method for interfacial discontinuity of multi-phase fluids. In *18^{th} AIAA Computational Fluid Dynamics Conference*, 2007.

K. M. T. Kleefsman and A. E. P. Veldman. An improved volume-of-fluid method for wave impact. In *European Congress on Computational Methods in Applied Sciences and Engineering*, 2004.

S. Koshizuka, Y. Oka, and S. Kondo. A staggered differencing technique on boundary-fitted curvilinear grids for incompressible flows along curvilinear or slant walls. *Computational Mechanics*, 7:123–136, 1990.

B. Kreiss. Construction of a curvilinear grid. *Journal of Scientific and Statistical Computing*, 4(2):270–279, 1983.

D. Kuzmin, R. Lohner, and S. Turek. *Flux–Corrected Transport: Principles, Algorithms and Applications*. Springer-Verlag, 2005.

P. Lax and B. Wendroff. Systems of conservation laws. *Communications on Pure and Applied Mathematics*, 13:217–237, 1960.

B. Leonard. Simple high-accuracy resolution program for convective modeling of discontinuities. *International Journal for Numerical Methods in Fluids*, 8:1291–1318, 1988.

B. Leonard. The ULTIMATE conservative difference scheme applied to unsteady one-dimensional advection. *Computer Methods in Applied Mechanics and Engineering*, 88: 17–74, 1991.

X. Liu, S. Osher, and T. Chan. Weighted essentially non-oscillatory schemes. *Journal of Computational Physics*, 115:200–212, 1994.

R. Lohner, C. Yang, J. D. Baum, L. H., P. D., and C. M. Charman. The numerical simulation of strongly unsteady flow with hundreds of moving bodies. *International Journal for Numerical Methods in Fluid*, 31:113–120, 1999.

F. Losasso, R. Fedkiw, and S. Osher. Spatially adaptive techniques for level set methods and incompressible flow. *Computers and Fluids*, 35:995–1010, 2006.

J. Mandal and J. Subramanian. On the link between weighted least-squares and limiters used in higher-order reconstructions for finite volume computations of hyperbolic equations. *Applied Numerical Mathematics*, 58:705–725, 2008.

D. J. Mavriplis. Unstructured grid techniques. *Annual Review of Fluid Mechanics*, 29:473–514, 1997.

D. Mavriplis. Multigrid solution of the two-dimensional Euler equations on unstructured triangular meshes. *AIAA Journal*, 26:824–838, 1988.

V. Mihalef, D. Metaxas, and M. Sussman. Textured liquids based on the marker level set. *Eurographics*, 26(3), 2007.

C. Min and F. Gibou. A second order accurate level set method on non-graded adaptive Cartesian grids. *Journal of Computational Physics*, 225:300–321, 2007.

K. R. Moyle and Y. Ventikos. Local remeshing for large amplitude grid deformations. *Journal of Computational Physics*, 227:2781–2793, 2008.

W. Mulder and S. Osher. Computing interface motion in compressible gas dynamics. *Journal of Computational Physics*, 100:209–229, 1992.

M. Murayama, K. Nakahashi, and M. K. Unstructured dynamic mesh for large movement and deformation. In *40^{th} AIAA Aerospace Sciences Meeting*, January 2002.

F. Mut, G. Buscaglia, and E. Dari. A new mass-conserving algorithm for level set redistancing on unstructured meshes. *Journal of Applied Mechanics*, 73:1011–1016, 2006.

B. Nichols and C. Hirt. Improved free surface boundary conditions for numerical incompressible–flow calculations. *Journal of Computational Physics*, 8:434–448, 1971.

A. Nishiguchi and T. Yabe. Finite-sized fluid particle in a nonuniform moving grid. *Journal of Computational Physics*, 47:297–302, 1982.

A. Nishiguchi and T. Yabe. Second-order fluid particle scheme. *Journal of Computational Physics*, 52:100–110, 1983.

W. F. Noh and P. Woodward. SLIC (Simple Line Interface Calculation) [for multifluid dynamics]. In *Proceedings of the 5th International Conference on Numerical Methods in Fluid Dynamics*, 1976.

N. N. Nourgaliev, T. Dinh, and T. Theofanous. Sharp treatment of surface tension and viscous stresses in multifluid dynamics. In *17th AIAA Computational Fluid Dynamics Conference*, 2005a.

R. R. Nourgaliev, S. Wiri, T. N. Dinh, and T. G. Theofanous. Adaptive strategies for mass conservation in level set treatment. In *17th AIAA Computational Fluid Dynamics Conference*, 2005b.

R. Nourgaliev, T. Dinh, and T. Theofanous. Adaptive characteristics-based matching for compressible multifluid dynamics. *Journal of Computational Physics*, 213:500–529, 2006.

S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. Springer, 2002.

S. Osher and J. A. Sethian. Fronts propagating with curvature-dependant speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 70:12–49, 1988.

S. Osher and C. W. Shu. High-order essentially nonoscillatory schemes for Hamilton-Jacobi equations. *Journal of Numerical Analysis*, 28(4):907–922, August 1991.

D. Peng, B. Merriman, S. Osher, H. Zhao, and M. Kang. A PDE-based fast local level set method. *Journal of Computational Physics*, 155:410–438, 1999.

J. Pilliod and E. Puckett. Second-order accurate volume of fluid algorithms for tracking material interfaces. *Journal of Computational Physics*, 199:465–502, 2004.

L. Qian, D. Causon, D. Ingram, and C. Mingham. Cartesian cut cell two-fluid solver for hydraulic flow problems. *ASCE Journal of Hydraulic Engineering*, 129:688–696, 2003.

W. Rider and D. Kothe. Reconstructing volume tracking. *Journal of Computational Physics*, 141:112–152, 1998.

P. Roache. *Verification and Validation in Computational Science and Engineering*. Hermosa,

1998.

P. Roe. Characteristic based schemes for the Euler equations. *Annual Review of Fluid Mechanics*, 18:337–365, 1986.

C. Roy. Review of code and solution verification procedures for computational simulation. *Journal of Computational Physics*, 205:131–156, 2005.

G. Russo and P. Smereka. A remark on computing distance functions. *Journal of Computational Physics*, pages 51–67, 2000.

A. Saruwatari, Y. Watanabe, and D. Ingram. Scarifying and fingering surfaces of plunging jets. *Coastal Engineering*, 56:1109–1122, 2009.

R. Scardovelli and S. Zaleski. Direct numerical simulation of free-surface and interfacial flow. *Annual Review of Fluid Mechanics*, 31:567–603, 1999.

J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93:1591–1595, February 1996.

J. A. Sethian and P. Smereka. Level set methods for fluid interfaces. *Annual Review of Fluid Mechanics*, 35:341–372, 2003.

J. Sethian. Evolution, implementation and application of level set and fast marching methods for advancing fronts. *Journal of Computational Physics*, 169:503–555, 2001.

C. W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. *Journal of Computational Physics*, 83:32–78, 1989.

C. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77:439–471, 1988.

J. Slater. Examining spatial (grid) convergence, Last visited June 2011. URL `www.grc.nasa.gov/WWW/wind/valid/tutorial/spatconv.html`.

S. Spekreijse. Multigrid solution of monotone second-order discretizations of hyperbolic conservation laws. *Mathematics of Computation*, 549(179):135–155, 1987.

M. Sun. *Numerical and Experimental Studies of Shock Wave Interation with Bodies*. PhD thesis, Tohoku University, 1998.

M. Sussman and D. G. Dommermuth. The numerical simulation of ship waves using Cartesian grid methods. In *Proceedings on 23rd Symposium of Naval Hydrodynamics*, 2000.

M. Sussman and E. Fatemi. An efficient, interface-preserving level set redistancing algorithm and its application to interfacial incompressible fluid flow. *SIAM Journal on Scientific Com-*

putation, 20(4):1165–1191, 1999.

M. Sussman and E. G. Puckett. A coupled level set and volume-of-fluid method for computing 3D and axisymmetric incompressible two-phase flow. *Journal of Computational Physics*, 162:301–337, 2000.

M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solution to incompressible two-phase flow. *Journal of Computational Physics*, 114:146–159, 1994.

M. Sussman, E. Fatemi, P. Smereka, and S. Osher. An improved level set method for incompressible two-phase flows. *Computers & Fluids*, 27(5-6):663–680, 1998.

M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, and M. Welcome. An adaptive level set approach for incompressible two-phase flows. *Journal of Computational Physics*, 148: 81–124, 1999.

P. Sweby. High resolution schemes using flux limiters for hyperbolic conservation laws. *Journal on Numerical Analysis*, 21(5):995–1011, 1984.

H. Takewaki and T. Yabe. The cubic-interpolated pseudo particle (CIP) method: Application to nonlinear and multi-dimensional hyperbolic equations. *Journal of Computational Physics*, 70:355–371, 1986.

H. Takewaki, A. Nishiguchi, and T. Yabe. Cubic interpolated pseudo-particle method (CIP) for solving hyperbolic-type equaitions. *Journal of Computational Physics*, 61:261–268, 1984.

J. Taylor, M. Rea, and D. Rogers. The Edinburgh curved tank. In *5th European Wave Power Conference*, 2003.

A. Tveito and R. Winther. *Introduction to Partial Differential Equations, A Computational Approach*. Springer-Verlag, 2005.

S. O. Unverdi and G. Tryggvason. A front-tracking method for viscous, incompressible, multi-fluid flows. *Journal of Computational Physics*, 100:25–37, 1992.

T. Utsumi, T. Kunugi, and T. Aoki. Stability and accuracy of the cubic interpolated propagation scheme. *Computer Physics Communications*, 101:9–20, 1997.

B. van Leer. Towards the ultimate conservative difference scheme. II. Monotonicity and conservation combined in a second-order scheme. *Journal of Computational Physics*, 14:361–370, 1974.

B. van Leer. Towards the ultimate conservative difference scheme. III. Upstream-centered finite-difference schemes for ideal compressible flow. *Journal of Computational Physics*, 23:263–275, 1977a.

B. van Leer. Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection. *Journal of Computational Physics*, 23:276–299, 1977b.

B. van Leer. Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method. *Journal of Computational Physics*, 23:101–136, 1979.

V. Venkatakrishnan. A perspective on unstructured grid flow solvers. Technical report, NASA, 1995a.

V. Venkatakrishnan. Convergence to steady state solutions of the euler equations on unstructured grids with limiters. *Journal of Computational Physics*, 118:120–130, 1995b.

Z. Wang and Z. Wang. The level set method on adaptive Cartesian grid for interface capturing. In *42nd AIAA Aerospace Sciences Meeting and Exhibition*, 2004.

Y. Watanabe and H. Saeki. Velocity field after wave breaking. *International Journal for Numerical Methods in Fluids*, 39:607–637, 2002.

Y. Watanabe, H. Saeki, and R. J. Hosking. Three-dimensional vortex structures under breaking waves. *Journal of Fluid Mechanics*, 545:291–328, 2005.

Y. Watanabe, A. Saruwatari, and D. Ingram. Free-surface flows under impacting droplets. *Journal of Computational Physics*, 227:2344–2365, 2008.

T. Yabe and T. Aoki. A universal solver for hyperbolic equations by cubic-polynomial interpolation, I. One-dimensional solver. *Computer Physics Communications*, 66:219–232, 1991.

T. Yabe, T. Ishikawa, P. Y. Wang, T. Aoki, Y. Kadota, and F. Ikeda. A universal solver for hyperbolic equations by cubic-polynomial interpolation, II. Two- and three-dimensional solvers. *Computer Physics Communications*, 66:233–242, 1991.

T. Yabe, Y. Zhang, and X. F. A numerical procedure - CIP - to solve all phases of matter together. In *Sixteenth International Conference on Numerical Methods in Fluid Dynamics*, 1998.

T. Yabe, H. Mizoe, K. Takizawa, H. Moriki, H. Im, and Y. Ogata. Higher-order schemes with CIP method and adaptive Soroban grid towards mesh-free scheme. *Journal of Computational Physics*, 194:57–77, 2004.

G. Yang, D. Causon, D. Ingram, R. Saunders, and P. Batten. A Cartesian cut-cell method for compressible flows – Part A: Static body problems. *Aeronautical Journal*, 101:47–56, 1997a.

G. Yang, D. Causon, D. Ingram, R. Saunders, and P. Batten. A Cartesian cut-cell method for compressible flows – Part B: Moving body problems. *Aeronautical Journal*, 101:57–65, 1997b.

G. Yang, D. Causon, and D. Ingram. Calculation of compressible flows about complex moving geometries using a three-dimensional Cartesian cut cell method. *International Journal for Numerical Methods in Fluids*, 33:1121–1151, 2000.

X. Yang, A. James, J. Lowengrub, X. Zheng, and V. Cristini. An adaptive coupled level-set/ volume-of-fluid interface capturing method for unstructured triangular grids. *Journal of Computational Physics*, 217:364–394, 2006.

H. Yee, R. Warming, and A. Harten. Implicit total variation diminishing (TVD) schemes for steady-state calculations. *Journal of Computational Physics*, 57:327–360, 1985.

W. Yue, C. Lin, and V. Patel. Numerical simulation of unsteady multidimensional free surface motions by level set method. *International Journal for Numerical Methods in Fluids*, 42: 853–884, 2003.

S. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics*, 31:335–362, 1979.

P. Zaspel and M. Griebel. GPU accelerated solver for the 3D two-phase incompressible Navier-Stokes equations. In *GPU Technology Conference*, 2010.

Y. Zhang and C. Shu. High-order schemes for Hamilton-Jacobi equations on triangular meshes. *Journal on Scientific Computing*, 24(3):1005–1030, 2003.

Y. Zhang and C. Shu. Third order WENO scheme on three dimensional tetrahedral meshes. *Communications in Computational Physics*, 5(2–4):836–848, 2009.